ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА, ИНФОРМАЦИОННЫЕ СИСТЕМЫ И ТЕХНОЛОГИИ

УДК 621.3.049.771.14

С.В. Гаврилов, Д.И. Рыжова

АЛГОРИТМ ОЦЕНКИ ПИКОВОГО ТОКА НА ЛОГИЧЕСКОМ УРОВНЕ ПРОЕКТИРОВАНИЯ НА ОСНОВЕ АНАЛИЗА РАСПРОСТРАНЕНИЯ ЛОГИЧЕСКИХ КОРРЕЛЯЦИЙ В СХЕМЕ

Для проектирования наноразмерных схем актуальна проблема оценки максимального тока, протекающего в цепях питания, который влияет на величину падения напряжения, или так называемого IR-drop эффекта. Известные подходы такие, как методы нижних оценок и методы генерации тестовых последовательностей, методы приближенных оценок пикового тока, не обеспечивают достаточно точное решение указанной задачи. Предлагаемый метод учета корреляций при анализе пикового тока предполагает совместное рассмотрение вещественных интервалов задержек распространения с булевыми интервалами входных векторов, для которых такая задержка достижима, и обеспечивает повышение достоверности оценки максимального тока за счет анализа логической совместимости для переключений вентилей.

Ключевые слова: *статический временной анализ (СТА), сложнофункциональный блок (СФ-блок), логические корреляции, анализ пикового тока.*

Введение. Проектирование современных микро- и наноэлектронных схем для технологий с проектными нормами ниже 90 нм сталкивается с важными задачами определения величины падения напряжения в шинах питания [1,2] (IR-drop эффект) и ширины шин питания для обеспечения работоспособности схемы. Нужно также учитывать, что максимальный ток, возникающий при скачке напряжения, приводит к понижению запаса помехоустойчивости схемы.

Промышленные программы такие, как Synopsys PrimeRail и Cadence Encounter, позволяют оценить ток потребления по результатам характеризации мощности, однако такая оценка является излишне пессимистичной. Программы генерации и моделирования тестовых последовательностей, например, Synopsys CustomSim и Cadence UltraSim, определяют нижнюю границу значения пикового тока; использование результатов таких программ может привести к недооценке величины падения напряжения в шинах питания и, соответственно, спровоцировать выход схемы из строя.

Цель работы. В данной статье предлагается метод оценки максимального тока потребления, обеспечивающий повышение достоверности результатов за счет отсева путей, которые не могут быть реализованы ни при одном наборе входных воздействий.

Существующие методы оценки нижней границы пикового тока. Методы моделирования тестовых последовательностей позволяют оценить величину тока потребления схемы снизу. Значение тока рассчитывается симуляторами как разность напряжений двух терминалов паразитного резистора, деленная на его сопротивление. Одними из основных производителей САПР, предоставляющих средства подготовки тестопригодных проектов, верификации и создания тестовых векторов, являются Cadence и Synopsys. Основа аппаратного проектирования в системе Cadence – платформа Incisive. Она пред-

ставляет собой единую среду программно-аппаратного проектирования, отладки, верификации и генерации тестов для цифроаналоговых СБИС. Средства Encounter [2] и UltraSim [3] являются современными инструментами для моделирования тестовых последовательностей. В компании Synopsys к средствам синтеза тестопригодных проектов можно отнести Design Compiler, DFT Compiler, BSD Compiler. Средство ATPG – TetraMAX ATPG, симуляторы – Nanosim [4], CustomSim [1]. Моделирование всех входных комбинаций схемы – процесс затратный и дорогостоящий, а в случае больших схем – невозможный из-за большой размерности задачи.

Другой подход к оценке пикового тока – моделирование тестовых последовательностей, полученных в результате применения генетических и эволюционных алгоритмов [5, 6]. Данные подходы широко применяются для решения множества комбинаторных задач по оптимизации, но не обеспечивают приемлемую точность оценки пикового тока в схеме, так как не гарантируют полноту тестового покрытия.

Существующие методы приблизительных оценок пикового тока. В статьях [7-10] верхняя граница максимального тока в шинах питания/земли оценивается с помощью распространения входных сигналов во временных интервалах через нормированные цепочки вентилей. Предлагаемый алгоритм, называемый «iMax», рассчитывает максимальный ток сигнала статистически в линейном времени с помощью независимой модели. Работа содержит следующие допущения: все входы – независимые, все первичные входы переключаются в момент времени t=0, схема – комбинационная, задержки вентилей фиксированы и известны заранее, форма сигнала – прямоугольный треугольник, различные параметры тока, такие как пик, длительность и фиксируемое время рассчитываются на этапе предварительной обработки из схемы на уровне параметров рассматриваемых и связанных с их входами и выходами вентилей. Другое предположение основано на том, что на стыке временных интервалов входы не меняются и, следовательно, нет соответствующего интервала неопределенности на выходе схемы.

Данный подход обеспечивает верхнюю оценку пикового тока, однако, оценка пикового тока должна учитывать и количество переключений, и количество узлов с большими выходными емкостями во время поиска максимума. Нахождение максимума само по себе недостаточно для обеспечения жесткой оценки пикового тока, так как необходимо учитывать также одновременное переключение нескольких вентилей.

Модели задержек библиотечных элементов. Существуют различные методики для моделирования поведения элемента при оценке задержки в статическом временном анализе. Традиционно для оценки задержки используется NLDM модель (Non-Linear Delay Model) [11, 12]. В NLDM модели задержка элемента моделируется с помощью таблиц соответствия. Если пренебречь активной составляющей сопротивления и полезной нагрузки ячейки, то задержка ячейки зависит от двух параметров – фронта входного сигнала и эффективной емкостной нагрузки выхода, определенной для технологического процесса, напряжения и температуры. В NLDM задержка ячейки моделируется и записывается в двумерные таблицы

$$D_{out}\left(S_{inp}^{k}, C_{out}^{l}\right),\ k \in \left[1 : N_S\right],\ l \in \left[1 : N_C\right],$$
$$S_{out}\left(S_{inp}^{k}, C_{out}^{l}\right),\ k \in \left[1 : N_S\right],\ l \in \left[1 : N_C\right]$$

для каждого переключения. Здесь D_{out} – выходной сигнал, S_{out} – длительность выходного фронта сигнала, S_{inp} – длительность фронта входного сигнала, C_{out} – емкость нагрузки.

Задержки элемента получаются из множественного SPICE моделирования для каждого множества S_{inp} и C_{out}. Поскольку таблица соответствия содержит только небольшой набор S_{inp} и C_{out}, интерполяции или экстраполяции необходимо оценить t_{delay} и S_{out} на желаемый набор Sin и Cout.

NLDM модель позволяет рассчитывать задержки быстро за счет снижения точности. Основные причины потери точности – интерполяция, вариация параметров и сложное поведение транзисторов. Однако из-за увеличения частоты и уменьшения технологического размера относительная погрешность, вносимая NLDM, больше не является приемлемой. Для того чтобы удовлетворить новым требованиям, компаниями Synopsys и Cadence были разработаны новые модели, которые заменяют NLDM. Были разработаны более сложные модели, в которых формы выходных сигналов могут быть получены более точно. Эти современные схемы моделирования уменьшили относительную погрешность расчета задержки. CCS модель (Composite current source) Synopsys и ECSM модель (Effective current source model) Cadence – наиболее известные стандарты моделирования интегральных схем.

CCS обеспечивает высокую точность расчета задержек ячеек, межсоединений, длительностей входного фронта и нагрузочной емкости при наличии паразитных элементов. Для дости-

жения необходимой точности вычислений задержек в CCS используются три модели: модель источника, модель приемника, редуцированная упорядоченная модель для расчета паразитных RC соединений.

Временная модель CCS состоит из двух частей.

1. Нелинейная модель источника переменного (во времени) тока, зависящего от напряжения:

$$I_{out}=F(t, S_{inp}, C_{out}),$$

где S_{inp} – длительность фронта входного сигнала, C_{out} – емкость нагрузки. Модель источника может использоваться как с моделью приемника, так и без нее.

График функции F имеет колоколообразную форму. В CCS модели функция выходного тока $I_{out}(t)$ запоминается в виде набора точек $\{t^k, I^k_{out}\}$ на интервале $[t_0, t_1]$. При пересечении порогового уровня $0.5 \cdot Vdd$ входным сигналом $V_{inp}(t)$ значение времени T_{ref} в модели запоминается.

2. Модели приемника, представляющего собой вход приемного элемента, описывающего, как меняется величина емкости в зависимости от формы (длительности) входного фронта и емкости на выходе. Для динамической корректировки значения емкости в течение перехода используются два значения емкости C_1 и C_2 (отдельные величины для переднего и заднего фронтов):

$$C\_1^{(r|f)}_{inp}\left(S_{inp}, C_{out}\right),\ C\_2^{(r|f)}_{inp}\left(S_{inp}, C_{out}\right),$$

где C_1 – это таблица значений эффективных входных емкостей, рассчитанных для первой половины фронта входного сигнала (от начала переключения до T_{ref}), а C_2 – это таблица значений емкостей, вычисленных для второй половины фронта входного сигнала (от T_{ref} до конца переключения).

В случае когда значение входной емкости слабо зависит от значения выходной емкости, остается зависимость моделей входных только от длительности входного фронта:

$$C\_1^{(r|f)}_{inp}\left(S_{inp}\right),\ C\_2^{(r|f)}_{inp}\left(S_{inp}\right).$$

Перед характеризацией проводятся измерения выходного тока как функции времени для указанной длительности входного фронта и выходной емкости. Используя токи и соответствующие емкости, восстанавливается форма сигнала напряжения. Для емкостей и формы входных фронтов, в которых не были проведены измерения тока, используется интерполяция.

Объем выходных данных для CCS модели значительно превышает объем результирующих данных для NLDM модели, однако количество моделирований электрической схемы в обоих случаях одинаково, при этом точность моделирования близка к полному моделированию на Spice.

В САПР фирмы Cadence применяется модель временных параметров вентилей, сходная с CCS моделью, – ECSM модель. ECSM является расширением формата Liberty, обеспечивающим способ хранения данных, которые совместимы с существующим форматом описания задержек.

ECSM – это метод вычисления задержек, который использует модель источника на основе тока и модель приемника с переменной емкостью выводов для точного вычисления задержки вентиля. Такой метод более эффективен для учета нелинейного поведения транзисторов при переключении и позволяет более точно моделировать межсоединения. В то время как CCS использует диаграммы токов явно, ECSM использует диаграммы напряжений, которые конвертируются в диаграммы токов во время расчета задержек:

$$V_{out}=G(t, S_{inp}, C_{out}),$$

где V_{out} – напряжение на выходе вентиля, S_{inp} – длительность фронта входного сигнала, C_{out} – емкость нагрузки.

Модель приемника ECSM записывается следующим образом:

$$C^{(r|f)}_{inp}\left(S_{inp}, C_{out}\right).$$

Значения функции $V_{out}(t)$ сохраняются в одномерные таблицы для интервала $[t_0, t_1]$, где точки t_0 и t_1 определяются условием $V_{out}(t) \in [0+\varepsilon, Vdd-\varepsilon]$, ε = const. При использовании диаграммы напряжений возможна более быстрая и точная характеризация библиотечных ячеек.

Теоретически модели CCS и ECSM модели драйверов эквивалентны, так как функции $I_{out}(t)$ и $V_{out}(t)$ связаны уравнением:

$$I_{out}(t) = C_{out} \cdot \frac{dV_{out}(t)}{dt}.$$

Практически, конечно, эти модели могут приводить к несколько различным результатам, так как функции $I_{out}(t)$ и $V_{out}(t)$ различаются областями определения. Кроме того, вычисление $I_{out}(t)$ из уравнения неизбежно сопровождается числовыми погрешностями.

Нужно отметить также, что благодаря масштабированию функций $V_{out}(t)$, ECSM модель может без изменения применяться для расчетов схем с различными значениями Vdd.

Модели NLDM, CCS и ECSM позволяют значительно увеличить точность логического моделирования интегральных схем, однако они

неприменимы к задаче оценки пикового тока в шинах питания по следующим причинам:

1) в результате предыдущих исследований было выявлено, что поведение тока зависит от значения смещения между фронтами: пиковый ток достигает максимального значения при значениях смещения, близких или равных нулю. Однако модели NLDM, CCS и ECSM не учитывают информацию о близких или одновременных переключениях входов вентилей;

2) NLDM, CCS и ECSM модели хранят информацию только о выходных токах схемы, в общем же случае, значение выходного тока не является максимальным значением тока потребления в схеме;

3) NLDM, CCS и ECSM модели не позволяют моделировать IR-drop эффект.

Существующие модели тока. Из публикаций известны упрощенные модели тока, представляющие кривые тока от времени в форме прямоугольника [8] (рисунок 1), треугольника [10] (рисунок 2), трапеции [7] (рисунок 3). Существуют также подходы, которые ориентированы на полное электрическое моделирование входных тестовых последовательностей для получения гладкой кривой, отражающей реальное поведение тока потребления (рисунок 4).

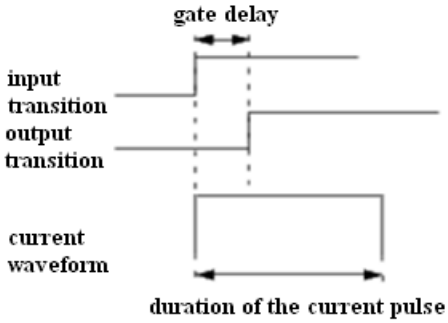


Рисунок 1 – Прямоугольная форма сигнала тока

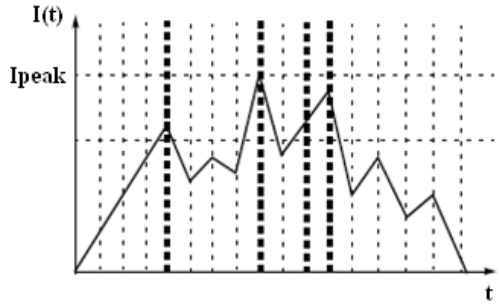


Рисунок 2 – Треугольная форма сигнала тока

Для повышения точности моделей в качестве прототипа для разрабатываемого подхода были выбраны CCS и ECSM модели, которые используются для других целей. Они сохраняют в виде таблиц значения функций I_{out} и V_{out} от времени соответственно, но не хранят информацию о зависимостях тока в шине питания от времени:

$$I_{out} = F\left(t, S_{inp}, C_{out}\right),$$
$$V_{out} = G\left(t, S_{inp}, C_{out}\right).$$

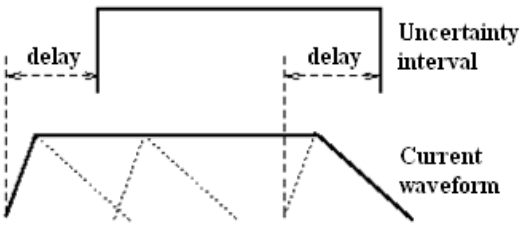


Рисунок 3 – Трапециевидная форма сигнала тока

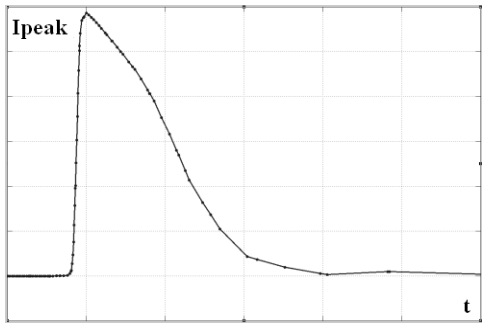


Рисунок 4 – Гладкая кривая тока

В связи с этим появилась необходимость разработки новой модели, позволяющей сохранять значения тока потребления в контрольных точках, которые выбираются по уровням входных и выходных напряжений [13]:

$$I_o = \{(t_i, I_i) : V_o(t_i) = OL_i \cdot V_{dd}\},$$
$$I_1 = \{(t_i, I_i) : V_1(t_i) = IL_i \cdot V_{dd}\},$$
$$I_2 = \{(t_i, I_i) : V_2(t_i) = IL_i \cdot V_{dd}\},$$

где $IL = \{IL_i\} = \{0.0\ 0.1\ 0.2\ 0.5\ 0.8\ 1.0\}$ – уровни входных напряжений, $OL = \{OL_i\} = \{0.01\ 0.1\ 0.2\ 0.5\ 0.8\ 0.9\ 0.99\ 0.999\}$ – уровни выходного напряжения.

Полученная модель тока потребления сохраняет информацию для NLDM модели:

$$D_1 = \left(t_i : V_o(t_i) = 0.5 \cdot V_{dd}\right) - \left(t_j : V_1(t_j) = 0.5 \cdot V_{dd}\right),$$
$$D_2 = \left(t_i : V_o(t_i) = 0.5 \cdot V_{dd}\right) - \left(t_j : V_2(t_j) = 0.5 \cdot V_{dd}\right),$$
$$S_o^r = \left(t_i : V_o(t_i) = 0.8 \cdot V_{dd}\right) - \left(t_j : V_o(t_j) = 0.2 \cdot V_{dd}\right).$$

Сравнение результатов выбранного метода с NLDM моделью показало, что ошибка при нахождении максимального значения тока не превышает 3%.

Для того чтобы сократить число запусков моделирования для оценки пикового тока в больших схемах, был разработан алгоритм характеризации тока в контрольных точках на основе двоичного деления по параметру смещения входных фронтов (рисунок 5). Из полученных значений тока, фронтов и смещений была со-

ставлена таблица характеризации, которую в дальнейшем можно использовать в алгоритме интерполяции по смещениям между входными фронтами.

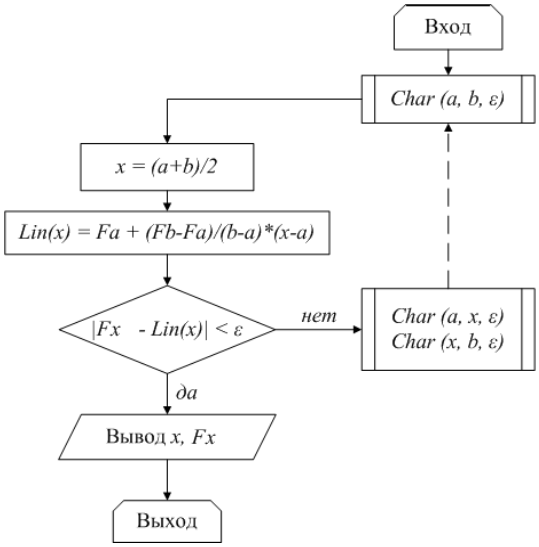


Рисунок 5 – Блок-схема алгоритма характеризации тока в контрольных точках

Согласно классическому подходу к билинейной интерполяции каждому значению входного фронта соответствует одно значение смещения, в нашем случае каждому значению входного фронта соответствует четыре значения смещения, поэтому необходим новый подход к интерполяции токов в контрольных точках (рисунок 6), так называемая трилинейная интерполяция [14, 15].

На первом шаге загружается таблица характеризации с значениями тока в контрольных точках. На втором шаге в программу вводятся значения входных фронтов и смещения между ними, для которых нужно определить максимальное значение тока потребления. В зависимости от того, в какой интервал попали указанные значения фронтов, возможны следующие варианты интерполяции:

1) если оба введенных фронта совпали с фронтами из таблицы характеризации, то проводится линейная интерполяция по значению смещения между входными фронтами;

2) если только один из фронтов совпал с фронтом из таблицы характеризации, то проводится билинейная интерполяция по значениям смещения и входного фронта, неопределенного в таблице характеризации;

3) если оба введенных фронта не совпали с фронтами из таблицы характеризации, проводится линейная интерполяция по значению смещения между входными фронтами и далее билинейная интерполяция по значениям фронтов входных сигналов.

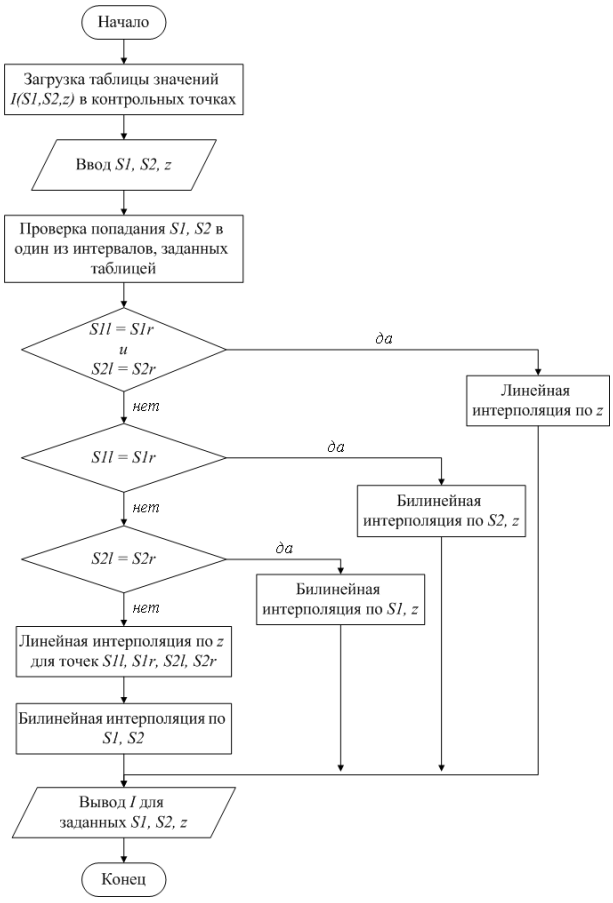


Рисунок 6 – Блок-схема алгоритма трилинейной интерполяции пикового тока

На заключительном этапе в итоговый файл выводятся значения тока в указанных контрольных точках для заданных значений фронтов и смещения между ними.

В сложных КМОП-схемах (сотни вентилей) для оценки максимального тока потребления используется следующая методика суммирования кривых [16]:

— определяются интервалы временных окон переключений логических вентилей (например, с помощью программы Synopsys PrimeTime);

— для каждого временного интервала значения токов вентилей, которые переключаются в одном временном окне, суммируются. По результатам строится карта токов (или огибающая):

$$I_j = \max_p \sum_i I_{ijp}, j \in [1:100].$$

Для повышения достоверности оценки пикового тока в шинах питания необходим усовершенствованный подход, включающий анализ логической совместимости для переключения вентилей.

Адаптация метода резолюций для повышения достоверности анализа пикового тока. В статьях [17, 18] предлагается метод анализа шумов в цифровых КМОП-схемах на основе

распространения логических корреляций. Метод анализа помех с применением правила резолюций основан на следующих идеях.

1. Формирование списка узлов, в которых нужно определить уровень помех.

2. Формирование исходного списка ограничений из описания КМОП-схемы на транзисторном уровне.

3. Редукция системы логических ограничений на основе обобщенного метода исключений Гаусса.

4. Распространение и генерация простых логических импликаций на основе метода резолюций, что в итоге позволяет значительно ускорить применение метода резолюций и свести экспоненциальную задачу вывода ограничений к упрощенной задаче вывода импликаций линейной или квадратичной сложности.

Применительно к проблеме анализа пикового тока описанный выше подход модифицируется согласно задаче нахождения запретов (логических ограничений) на одновременное переключение вентилей на основе метода резолюции [19, 20].

Правило резолюций – это правило вывода булевых соотношений, исходя из метода доказательства истинности или ложности предположения от противного. Пусть задана пара (*V,R*), где *V={a,b,...}* – множество булевых переменных; а *R* – множество булевых соотношений типа *A=B*; *A,B* – выражения на множестве *V*. Согласно классической версии множество R преобразуется в соотношение f=1, где f представляет собой конъюнктивную нормальную форму (произведение сумм литералов, где литерал – это либо переменная, либо ее отрицание). Правило резолюции в классическом понимании можно записать следующим образом:

$$a + B = 1, \quad \bar{a} + C = 1 \quad \mapsto \quad B + C = 1,$$

где *B, C* – суммы литералов.

Существует также модификация метода резолюций, в котором множество R преобразуется в соотношение g=0, где g – это дизъюнктивная нормальная форма (сумма произведений литералов). Данный подход напрямую связан с методом импликаций.

Простая логическая импликация (ПЛИ) между двумя узлами схемы *x, y* – это отношение типа $(x = v) \Rightarrow (y = u)$ можно использовать обозначение $x^v \Rightarrow y^u$, где x^v, y^u – это литералы:

$$x^v = \begin{cases} \bar{x} & при \quad \mathrm{v} = 0, \\ x & при \quad \mathrm{v} = 1. \end{cases}$$

Отношение $a \Rightarrow b$ является отношением частичного порядка, т.е. следующие соотношения равносильны:

$$a \Rightarrow b \Leftrightarrow a \le b \Leftrightarrow a \cdot \bar{b} = 0$$

и выполняются следующие законы:

— рефлексивность $a \Rightarrow a$,

— антисимметричность: если $a \Rightarrow b$ и $b \Rightarrow a$, то $a = b$,

— транзитивность: если $a \Rightarrow b$ и $b \Rightarrow c$, то $a \Rightarrow c$.

Еще одно важное свойство импликаций – это контрапозитивный закон, вытекающий из закона двойного отрицания:

$$a \Rightarrow b \Leftrightarrow a \cdot \bar{b} = 0 \Leftrightarrow \bar{b} \cdot \bar{\bar{a}} = 0 \Leftrightarrow \bar{b} \Rightarrow \bar{a}.$$

Транзитивный и контрапозитивный законы можно использовать для генерации новых импликаций. Набор соответствующих правил вывода новых ПЛИ выглядит следующим образом:

$$a \Rightarrow b \quad \mapsto \quad \bar{b} \Rightarrow \bar{a},$$
$$a \Rightarrow b, \; b \Rightarrow c \quad \mapsto \quad a \Rightarrow c.$$

В качестве другого механизма вывода новых ПЛИ можно использовать распространение ПЛИ через простые вентили.

$$\forall a, b \in B : a \cdot b \le a, a \cdot b \le b,$$
$$\forall a, b \in B : a \le a + b, b \le a + b.$$

Исходя из приведенных соотношений, можно сделать следующие выводы:

1) если $c = a \cdot b$, то $c \le a$, $c \le b$, т.е. $c \Rightarrow a$, $c \Rightarrow b$, $\bar{a} \Rightarrow \bar{c}$, $\bar{b} \Rightarrow \bar{c}$;

2) если $c = a + b$, то $a \le c$, $b \le c$, т.е. $c \Rightarrow a$, $c \Rightarrow b$, $\bar{a} \Rightarrow \bar{c}$, $\bar{b} \Rightarrow \bar{c}$;

3) если $c = a \cdot b$, $x \le a$, $x \le b$, то $x \le c$, т.е. $x \Rightarrow c$, $c \Rightarrow \bar{x}$,

4) если $c = a + b$, $a \le x$, $b \le x$, то $c \le x$, т.е. $c \Rightarrow x$, $\bar{x} \Rightarrow \bar{c}$.

Отсюда вытекают правила вывода при распространении ПЛИ от входов вентиля к выходам:

1) $c = a \cdot b$, $x \Rightarrow \bar{a} \quad \mapsto \quad x \Rightarrow \bar{c}$;

2) $c = a \cdot b$, $x \Rightarrow \bar{b} \quad \mapsto \quad x \Rightarrow \bar{c}$;

3) $c = a + b$, $x \Rightarrow a \quad \mapsto \quad x \Rightarrow c$;

4) $c = a + b$, $x \Rightarrow b \quad \mapsto \quad x \Rightarrow c$;

5) $c = a \cdot b$, $x \Rightarrow a, x \Rightarrow b \quad \mapsto \quad x \Rightarrow c$;

6) $c = a + b$, $x \Rightarrow \bar{a}, x \Rightarrow \bar{b} \quad \mapsto \quad x \Rightarrow \bar{c}$.

И правила вывода при распространении ПЛИ от выходов вентиля к входам:

1) $c = a \cdot b$, $x \Rightarrow c \mapsto x \Rightarrow a, x \Rightarrow b$;

2) $c = a + b$, $x \Rightarrow \bar{c} \mapsto x \Rightarrow \bar{a}, x \Rightarrow \bar{b}$.

Оценка пикового тока в терминах MWIS проблемы. По результатам анализа распространения логических корреляций вдоль схемы можно построить граф парных ограничений – это тройка $G=(V,E,w)$, где V – набор вершин, E – набор ребер, $E\subseteq\{\{u,v\}:u,v\in V,u\neq v\}$, и w – весовая функция вершин такая, что

$$w(u)\geq 0,\ \forall u\in V,$$
$$w(K)=\sum_{u\in K}w(u),$$

где $K\subseteq V$.

Граф ограничений строится на основе простых логических импликаций (ПЛИ) для каждого анализируемого пути в схеме. Узлы в графе парных ограничений являются выходами вентилей схемы. Узлы, в которых возникает пиковый ток, называются узлами-агрессорами А. Ребро $e=(a_i,a_j)\in E_C$ соответствует ПЛИ $a^{\alpha_i}a^{\beta_j}$, которая является запретом на одновременное влияние агрессоров a_i и a_j на максимальный ток потребления схемы. Таким образом, ребро между узлами в графе добавляется в том случае, если узлы переключаются в одном направлении и имеют логическое ограничение на состояние 00 или 11. Кроме того, ребра в графе строятся между узлами, которые переключаются в разных временных интервалах.

В результате задача оценки максимального тока потребления сводится к решению задачи нахождения максимального взвешенного независимого набора вершин (МВННВ), или MWIS (Multiple Weighted Independent Set) задачи [21, 22]. Задача нахождения МВННВ на графе является NP-полной, однако из публикаций известны её эффективные решения при некоторых ограничивающих условиях [23].

Пусть N(u) и N(K) – это множества соседних вершин для $u\in V$ и $K\subseteq V$ соответственно, тогда:

$$N(u)=\{v:\{u,v\}\in E\},N(K)=\bigcup_{u\in K}N(u).$$

Получим, что набор вершин $K\subseteq V$, удовлетворяющий условию $N(K)\cap K=\varnothing$, называется независимым. Пусть $\Omega[G]$ является множеством всех независимых наборов вершин графа парных ограничений $G=(V,E,w)$. Тогда МВННВ будет называться независимое множество вершин B* такое, что:

$$w(B^*)=\max\{w(S):S\in\Omega[G]\}.$$

В случае использования всех, а не только парных ограничений, можно построить гиперграф, в котором ребра (гипер-ребра) могут соединять более двух вершин (рисунок 7).

Задача нахождения МВННВ в гиперграфе является трудоемкой, поэтому для получения нижней границы оценки пикового тока можно заменить каждое гипер-ребро парой обычных ребер (рисунок 8). Однако такое представление не отражает реальной картины исходных противоречий. Для получения оценки максимального тока сверху следует оставить в графе только парные ограничения (ПЛИ).

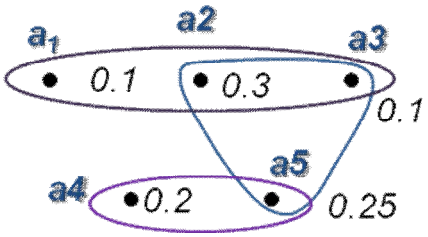


Рисунок 7 – Ограничения/гипер-ребра: {$a_1a_2a_3$, a_4a_5, $a_2a_3a_5$}; МВНН = {a_1,a_2,a_5}, w=0.65

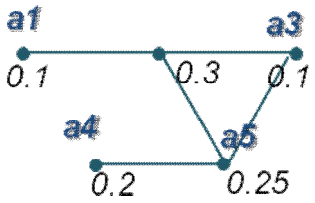


Рисунок 8 – Ограничения (ПЛИ) ребра: {a_1a_2, a_2a_3, a_3a_5, a_2a_5, a_4a_5, a_1a_3}; МВНН = {a_1, a_4}, w=0.35

Анализ пикового тока на основе метода ветвей и границ. Метод ветвей и границ – это метод для нахождения оптимальных решений задач оптимизации [24], модификация метода полного перебора с отсевом подмножеств допустимых решений, заведомо не содержащих оптимальных решений.

Алгоритм метода ветвей и границ на примере поиска минимума функции *f(x)*содержит следующие процедуры.

1. Ветвление. На этом этапе множество допустимых значений переменной *x* разбивается на подмножества, в каждом из которых находится экстремальное значение функции. Процедура рекурсивно применяется к подмножествам, в результате из подмножеств строится дерево поиска.

2. Нахождение оценок. На данном этапе осуществляется поиск верхних и нижних границ решения задачи на подмножестве допустимых значений переменной *x*.

Для того чтобы оценить пиковый ток с помощью метода ветвей и границ, необходимо обнулить начальное значение тока потребления и подать набор тестов, который задействует все узлы из взвешенного независимого набора вершин. В том случае, если очередная тестовая последовательность приведет к увеличению значе-

ния тока потребления, то найденная оценка автоматически становится максимальной. Алгоритм проходит по всему множеству узлов и заканчивает работу.

Заключение. По результатам анализа существующих подходов к оценке пикового тока разработана улучшенная модель тока на основе метода контрольных точек, который обеспечивает достаточно точное соответствие полученных для логического уровня максимальных значений тока, задержек и фронтов результатам схемотехнического моделирования. Для уменьшения размерности задачи моделирования на логическом уровне разработан алгоритм характеризации пикового тока в контрольных точках на основе дихотомического деления по параметру смещения входных фронтов, который позволяет в десятки раз ускорить моделирование СФ-блока. Предложен алгоритм трилинейной интерполяции токов потребления в контрольных точках, использующий результаты характеризации.

Предложен более точный подход к оценке максимального тока потребления, учитывающий логическую совместимость для переключений вентилей схемы. Разработана модификация метода ветвей и границ, ориентированная на поиск максимального взвешенного независимого набора вершин для оценки пикового тока в КМОП-схемах. Предложенные алгоритмы позволяют с приемлемой точностью и достаточно быстро оценить пиковый ток в схеме на логическом уровне проектирования.

Библиографический список

1. *Geden B.* Understand and Avoid Electromigration (EM) & IR-drop in Custom IP Blocks // Synopsys Webinars. 2011.
2. Encounter Power System // Cadence Data Sheet. 2012.
3. *Wei-Si Jiang*. An Effective EM/IR-drop Flow with UltraSim // CDNLive Silicon Valley. 2007.
4. NanoSim. Memory and Mixed-Signal Verification // Synopsys Data Sheet. 2003.
5. *Liu Y.-L., Wang C.-Y., Chen Y.-C.* A Novel ACO-based Pattern Generation for Peak Power Estimation in VLSI Circuits // Quality of Electronic Design. 2009. P. 317-323.
6. *Hsiao M.S.* Peak Power Estimation Using Genetic Spot Optimization for Large VLSI Circuit // Design, Automation and Test in Europe Conference. 1999. P. 175-179.
7. *Chowdhury S., Barkatullah J.S.* Estimation of maximum currents in MOS IC logic Circuits // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 1990. P. 642-654.
8. *Jiang Y.-M., Krstic A., Cheng K.-T.* Estimation for Maximum Instantaneous Current Through Supply Lines for CMOS Circuits // IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2000. P. 61-73.
9. *Kriplani H., Najm F., Hajj I.* Pattern Independent Maximum Current Estimation in Power and Ground Buses of CMOS VLSI Circuits: Algorithms, Signal Correlations and Their Resolution // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 1995. P. 998-1012.
10. Devadas S., Keutzer K., White J. Estimation of power dissipation in CMOS combinational circuits using Boolean function manipulation // IEEE Transactions on Computer-Aided Design. 1992. P. 373-383.
11. *Гаврилов С.В., Гудкова О.Н., Скачкова Е.П., Муханюк Н.Н., Соловьев Р.А*. Методы ускоренной характеризации больших параметризованных сложно-функциональных блоков // IV Всероссийская научно-техническая конференция «Проблемы разработки перспективных микроэлектронных систем – 2010»: сб. научн. тр. / под общей ред. Стемпковского А.Л. 2010. С. 154-159.
12. *Стемпковский А.Л., Гаврилов С.В., Глебов А.Л.* Методы логического и логико-временного анализа цифровых КМОП БИС. М: Наука. 2007. С. 36-48, 77-79,159-168.
13. *Рыжова Д.И., Гаврилов С.В., Щелоков А.Н.* Анализ пикового тока на основе результатов характеризации реальных библиотек логических вентилей // Сб. трудов Международного конгресса по интеллектуальным системам и информационным технологиям – 2013. Интеллектуальные САПР. С. 251-252.
14. *Гаврилов С.В., Рыжова Д.И., Стемпковский А.Л.* Проблема анализа пикового тока при проектировании сверхбольших интегральных схем на логическом уровне и современные методы ее решения // Информационные технологии. 2014. № 6. С. 58-63.
15. *Гаврилов С.В., Рыжова Д.И., Стемпковский А.Л.* Методы повышения точности оценки пикового тока на логическом уровне на основе анализа логических корреляций // Известия ЮФУ. Технические науки. 2014. № 7. С. 66-75.
16. *Sudhakar B., Hajj N. I.* Estimation of maximum current envelope for power bus analysis and design // International symposium on Physical design. 1998.
17. *Glebov A., Gavrilov S., Blaauw D.* False-Noise Analysis using Logic Implications // ICCAD. 2001. P. 515-520.
18. *Гаврилов С.В.* Методы анализа логических корреляций для САПР цифровых КМОП СБИС. М.: Техносфера. 2011. С. 96-115.
19. *Chang C.L., Lee R.C.T.* Symbolic Logic and Mechanical Theorem Proving // New York: Acad. Press. 1973.
20. *Чень Ч., Ли Р.* Математическая логика и автоматическое доказательство теорем. М.: Наука. 1983. 360 с.
21. *Papageorgiou D.J., Salpukas M.R.* The Maximum Weight Independent Set Problem for Data Association in Multiple Hypothesis Tracking // 8th International Conference on Cooperative Control and Optimization. 2009. P. 235-255.
22. *Brendel W., Amer M.* Multiobject tracking as maximum weight independent set // IEEE Conf. on Com-

puter Vision and Pattern Recognition. 2011. P. 1273-1280.

23. *Loukakis E., Tsouros C.* An Algorithm for the Maximum Internally Stable Set in a Weighted Graph // Intern. J. Computer Math. 1983. Vol. 13. P. 117-129.

24. *Shepard K.L.* Design methodologies for noise in digital integrated circuits // Proc. DAC. 1998. P. 94-99.

УДК 004.722

А.Ю. Романов, С.Р. Тумковский, Г.А. Иванова

МОДЕЛИРОВАНИЕ СЕТЕЙ НА КРИСТАЛЛЕ НА ОСНОВЕ РЕГУЛЯРНЫХ И КВАЗИОПТИМАЛЬНЫХ ТОПОЛОГИЙ С ПОМОЩЬЮ СИМУЛЯТОРА OCNS

Представлен обзор методов моделирования сетей на кристалле. Разработана высокоуровневая модель сети на кристалле на основе языка программирования Java, что позволило ускорить процесс моделирования на несколько порядков по сравнению с HDL моделями. Представлены результаты моделирования сетей на кристалле на основе регулярных и квазиоптимальных топологий с количеством до 100 узлов.

Ключевые слова: *сеть на кристалле (СтнК), моделирование СтнК, СтнК на основе квазиоптимальных топологий, СтнК на основе регулярных топологий, модель OSI.*

Введение. Методы моделирования сетей на кристалле. Одним из наиболее распространенных методов, используемых для моделирования сетей на кристалле (СтнК), является применение HDL языков описания аппаратуры (например, SystemVerilog) и средств событийного моделирования (например, ModelSim). Это позволяет получить точную высокопараметризированную модель СтнК, примером которой является Netmaker [1, 2]. Основной недостаток данного подхода заключается в слишком долгом процессе моделирования: только на один цикл моделирования в Netmaker СтнК с 9 узлами необходимо более 2 часов, и с увеличением количества узлов необходимое для моделирования время стремительно возрастает. Применение упрощенных HDL моделей СтнК (таких, как NoCSimp [3]) дает возможность на порядок сократить время моделирования, однако при моделировании СтнК с несколькими десятками узлов проблему не решает.

Согласно данным, приведенным в работе [4], максимальная скорость моделирования с помощью ModelSim составляет $3{,}2 \cdot 10^3$ цикл/с, SystemC – $20 \cdot 10^3$ цикл/с, в то время как чип FPGA функционирует с рабочей частотой от 50 МГц ($50 \cdot 10^3$ цикл/с). Развитием HDL моделирования является реализация сети в кристалле FPGA (прототипирование), где анализ осуществляется в реальном масштабе времени на уровне аппаратуры [4, 5]. Однако данный подход требует больших затрат времени на разработку проекта, наличия специального оборудования и применяется на поздних этапах проектирования.

Другой способ ускорения процесса моделирования связан с применением симуляции, которая представляет собой тестирование модели передачи данных в СтнК, описанной на языке высокого уровня. Так, в работе [6] представлена зависимость скорости параллельной обработки данных от параметров СтнК и анализируется влияние на нее задержек при передаче данных с увеличением размерности сети. В работе [7] описывается моделирование на сетях Petri в среде симулятора Visual Object Net – для анализа конкуренции, взаимодействия и конфликтов данных в коммуникационной среде СтнК. Интерес представляет также работа [8], где передача данных в СтнК представлена в виде набора параллельно выполняемых процессов, описанных на языке C/C++. Недостатком указанных подходов является использование ими упрощенной сетевой модели без учета всех параметров СтнК.

Отдельно следует упомянуть работу [9], в которой представлен мощный симулятор NOCsim на языке C/C++ (компиляция и компоновка отдельных модулей выполняется с помощью языка скриптов PERL), моделирующий Nostrum СтнК как множество потоков данных, проходящих через компоненты коммуникационной среды СтнК. К сожалению, данная модель не учитывает всех особенностей СтнК и не спо-

собна выполнять моделирование нерегулярных топологий, не содержит удобного графического интерфейса, а компоновка и компиляция проекта осложнены необходимостью применения специализированных скриптов.

Еще одним высокоуровневым универсальным программным симулятором на языке Java является gpNoCsim. В работе [10] приведены его описание и результаты моделирования различных регулярных топологий. К сожалению, введение поддержки новой топологии в симуляторе gpNoCsim требует полной переработки его архитектуры.

Цель работы – создание программного симулятора, который даст возможность ускорить процесс моделирования и проанализировать топологии с количеством узлов до 100 без потери точности в сравнении с HDL моделированием.

Разработка программной модели СтнК. Известно, что в архитектурных решениях СтнК используются концептуальные принципы построения компьютерных сетей. Организация внутренней многоуровневой подсистемы связи между IP блоками СтнК подобна базовой модели OSI, которая была взята за основу при разработке симулятора On-Chip Network Simulator (OCNS) [11]. На рисунке 1 представлен процесс передачи данных по сети при обмене сообщениями между двумя IP блоками, а также приведено соответствие уровней модели OSI уровням представления СтнК [12].

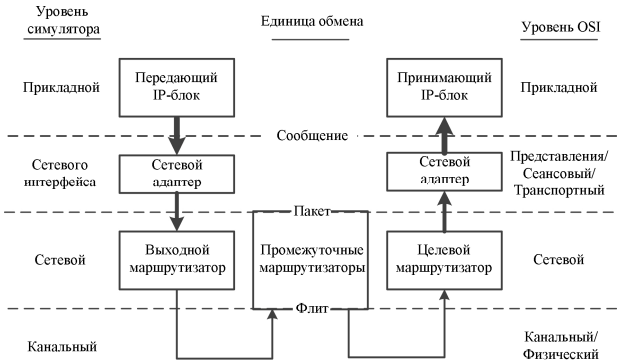


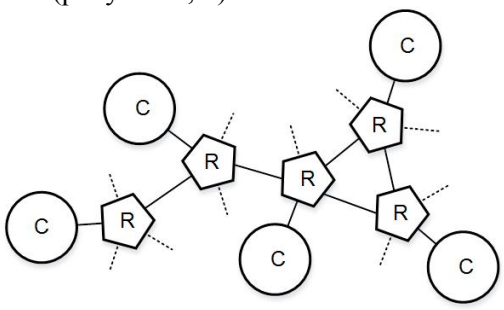
Рисунок 1 – Сопоставление структуры симулятора OCNS и сетевой модели OSI

Функция IP блоков состоит в генерации сетевого трафика, управление которым осуществляется с помощью маршрутизаторов. Кроме того, IP блоки и маршрутизаторы выполняют сбор статистической информации, которая впоследствии используется для сравнения параметров различных топологий СтнК. Поскольку на начальном этапе моделирования производится сбор статистических данных, которые не отражают реальной картины происходящего, в симуляторе предусмотрена поддержка «warm-up» циклов, на протяжении которых статистические данные не накапливаются.

On Chip Network Simulator позволяет моделировать СтнК, структура маршрутизаторов которых аналогична той, что была предложена в работе [13], а управление потоком передачи данных выполняется с помощью wormhole технологии с поддержкой виртуальных каналов. Реализована также возможность моделирования GALS сетей, для которых характерно наличие нескольких временных доменов. При моделировании предполагается, что все маршрутизаторы сети тактируются от источников синхронизации с одинаковой частотой, но со случайным смещением по фазе.

В базовом исполнении OCNS использовался для проведения моделирования ограниченного количества регулярных топологий [11]. Добавление новой топологии требовало изменения модуля маршрутизации и конфигурации СтнК, поэтому в дальнейшем [14] структура симулятора была переработана: каждый маршрутизатор получил таблицы маршрутизации, а топологию СтнК стало возможным задавать с помощью матрицы связей между маршрутизаторами. Это позволило проводить моделирование произвольных топологий, включая и квазиоптимальные.

Структуру модели СтнК On-Chip Network Simulator можно представить на нескольких уровнях (рисунки 2, 3).



C – ядро с сетевым интерфейсом;
R – маршрутизатор

Рисунок 2 – Верхний уровень модели СтнК с нерегулярной топологией

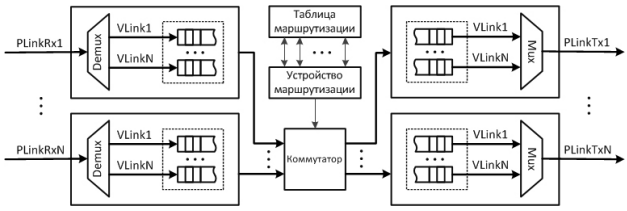


Рисунок 3 – Модель СтнК на уровне маршрутизатора

За каждый уровень модели, в соответствии с уровнями представления сетевой модели OSI (рисунок 1), отвечает собственный модуль. Мо-

дуль верхнего уровня иерархии TContollerOCNS руководит процессом моделирования. Модуль TNetworkManager строит сеть в соответствии с параметрами входного файла. Модуль TNetwork отвечает за перемещение трафика через сеть, за обновление состояний маршрутизаторов и за сбор статистики. TCore симулирует работу IP-ядра, т. е. осуществляет генерацию сообщений с заданной частотой, дробление их на флиты и отправку маршрутизатору, принятие флитов, восстановление исходного сообщения, сбор статистики. TRouter моделирует работу маршрутизатора и содержит набор контроллеров соединений TPLinkRx/TPLinkTx, таблицы маршрутизации и схемы коммутации. Каждый из TPLinkRx/TPLinkTx содержит буфер TBufferRx/TBufferTx. В свою очередь, TBufferRx содержит информацию для коммутации Data-флитов, передающихся вслед за Head-флитом.

Параметры модели задаются с помощью конфигурационного xml файла. Результаты моделирования выводятся в диалоговое окно, а выбранные параметры сохраняются в сводной таблице. Симулятор позволяет последовательно выполнять несколько итераций моделирования с разной конфигурацией.

После моделирования симулятор представляет следующие параметры производительности СтнК: количество отправленных и принятых пакетов; среднее время доставки пакета; среднее количество промежуточных сегментов, которые проходит пакет (количество переходов); пропускную способность маршрутизатора; пропускную способность сети; среднюю загрузку физического канала связи; загруженность входящих и исходящих буферов маршрутизатора и IP-блока и др.

Модель OCNS реализована на языке программирования Java с использованием Qt Jambi Framework, что обеспечивает все преимущества объектно-ориентированного программирования, кроссплатформенности программных решений и скорости их разработки. Полная независимость уровней представления OCNS (рисунок 1) позволяет выполнять разработку, модификацию и апробацию различных моделей СтнК и их компонентов с минимальными затратами времени.

Важной чертой OCNS является осуществление поддержки различных способов генерации сетевого трафика: 1) путем установления приоритета выбора узлов назначения по принципу центральных или периферийных «горячих точек»; 2) искусственным формированием адреса назначения (случайным порядком, транспонированием, инверсией, побитовым смещением, по принципу «торнадо» и по принципу коммуникации с ближайшими соседями) [15]; 3) путем подключения стандартных тестовых последовательностей, полученных при исследовании реальных приложений [15, 16] (например, тестовая последовательность, которая эмулирует 802.11a WiFi приемник с 25 задачами; тестовая последовательность MPEG4 декодера с 12 задачами; E3S сетевой тест для 12 задач и т. п.).

Экспериментальные исследования. Результаты моделирования СтнК на основе регулярных и квазиоптимальных топологий с помощью OCNS. Моделирование mesh, torus, древовидной и квазиоптимальных топологий (количество связей – от 8 до 18 для 9 узлов; средняя длина пакета – 10 флитов; размер флита – 32 бита; интенсивность введения пакетов в сеть – неодинаковая у разных узлов) проводилось с применением разработанного симулятора OCNS. Применялись 5-портовые маршрутизаторы, каждый порт которых имел по 4 виртуальных канала размером в 4 флита; процесс моделирования происходил до тех пор, пока каждый узел не отправлял по 1100 пакетов и они не были приняты. Для достижения устойчивого процесса подсчет статистики каждым узлом начинался после поступления первых 100 warm-up пакетов.

Время моделирования одного варианта топологии не превышало 1 мин (см. таблицу), что было значительно меньше, чем при использовании низкоуровневых SystemVerilog моделей Netmaker и NoCSimp.

Затраты времени на моделирование с помощью различных моделей СтнК

Модель	Количество узлов	
	9	100
Netmaker [2]	>2 ч.	–
NoCSimp [3]	7–10 мин	–
OCNS [14]	<1 мин	5 мин

Результаты моделирования в полной мере совпадают с данными, полученными с помощью Netmaker (рисунок 4): в случае, когда количество соединений квазиоптимальной и регулярной топологий с одинаковым количеством узлов совпадает, у регулярной топологии квадратной формы пропускная способность выше на 2–4 %. То есть, использование высокоуровневой модели On Chip Network Simulator дает результаты моделирования, близкие к тем, которые получены при применении точной низкоуровневой HDL модели Netmaker, но за значительно меньшее время (примерно на 2 порядка) (см. таблицу). Кроме того, результаты моделирования топологий mesh и torus совпадают с данными, полученными независимо другими авторами с помощью

высокоуровневого симулятора СтнК gpNoCsim [10].

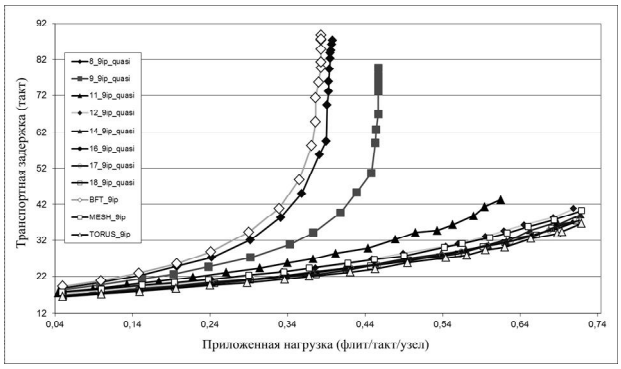


Рисунок 4 – Результаты моделирования регулярных и квазиоптимальных топологий с количеством связей от 8 до 18 для 9 узлов

Также было проведено моделирование квазиоптимальных топологий и прямоугольных топологий mesh и torus для 12 узлов (рисунок 5) [11].

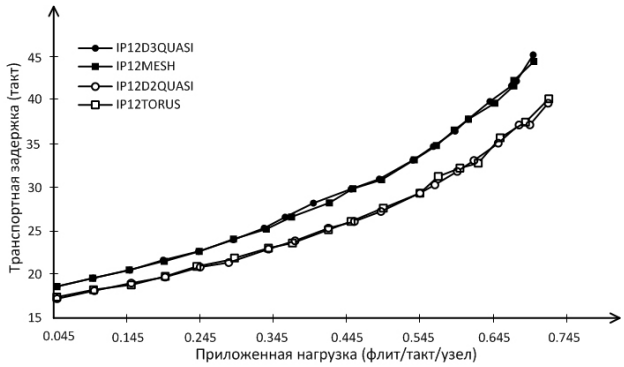
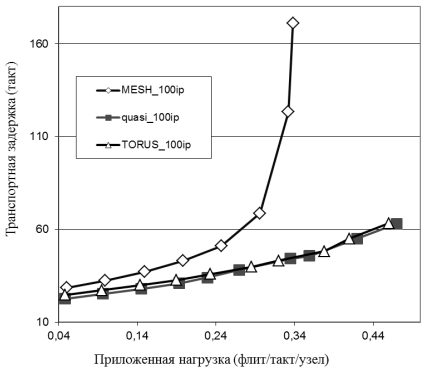


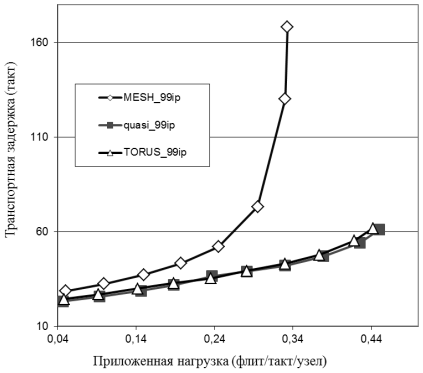
Рисунок 5 – Результаты моделирования топологий для 12 узлов

Квазиоптимальные топологии для 12 узлов имеют 21 соединение при диаметре 2 и 16 соединений – при диаметре 3, в то время как топология torus – 24 соединения при диаметре 3, а mesh – 17 соединений при диаметре 5. При этом графики зависимости задержки пакетов от интенсивности введения пакетов практически совпадают: у torus – с квазиоптимальной топологией диаметром 2 (IP12D2) и у mesh – с квазиоптимальной топологией диаметром 3 (IP12D3). То есть, согласно результатам моделирования, использование квазиоптимальных топологий позволило уменьшить затраты ресурсов без потери пропускной способности по сравнению с torus и mesh топологией прямоугольной формы на 3 и на 1 соединение (соответственно на 12,5 % и на 5,9 %).

Из-за больших затрат времени моделирование топологий, количество узлов которых достигает десятков, с помощью HDL подхода является проблематичным (см. таблицу). Однако генетический алгоритм GeNoC позволяет генерировать квазиоптимальные топологии, количество узлов которых достигает 100 [17]. Именно для моделирования таких СтнК и разработана OCNS. Для проведения сравнительного анализа СтнК на базе различных топологий с помощью генетического алгоритма GeNoC синтезированы топологии для количества узлов 49, 50, 99 и 100. Полученные квазиоптимальные топологии имеют меньшие показатели по диаметру и среднему расстоянию между узлами по сравнению с топологией torus (до 20 % и 19,4 % соответственно) и mesh (на 55,6 % и на 39 % соответственно) при незначительном различии в количестве соединений. Графики зависимостей задержки пакетов от интенсивности ввода их в СтнК, полученные в результате моделирования указанных выше топологий в OCNS, приведены на рисунках 6 и 7. Параметры моделирования были оставлены без изменений. Максимальная продолжительность полного цикла моделирования при различных интенсивностях введения пакетов была получена для топологии mesh со 100 узлами и не превысила 5 минут.



а



б

Рисунок 6 – Результаты моделирования топологий для 99 (а) и 100 (б) узлов

Для удобства сравнения пропускной способности топологий выбран их порог насыщения на уровне средней задержки прохождения пакетов, равной 60 тактов (минимальная задержка для топологий составляет 20 тактов).

Проведена сплайн-интерполяция графиков функций, и найдено значение пропускной способности при пороговом значении задержки пакетов в 60 тактов.

Как видно из рисунка 6, графики зависимости транспортной задержки от приложенной нагрузки для квазиоптимальной и torus топологий практически совпадают, в то время как топология mesh имеет значительно худшую пропускную способность. При N = 100 (рисунок 6,б) пропускная способность квазиоптимальной топологии составляет 0,440, топологии torus – 0,429 (на 2,1 % меньше), а топологии mesh – 0,279 (на 35 % меньше) [14]. При этом квазиоптимальная топология имеет на 8 соединений меньше, чем топология torus и только на 12 больше, чем топология mesh. Для топологий с 99 узлами (рисунок 6,а) ситуация практически не изменилась – пропускная способность увеличилась на 2–3 % за счет меньшего количества узлов, поскольку геометрические размеры регулярных топологий 11x9 узлов близки к квадратной оптимальной форме.

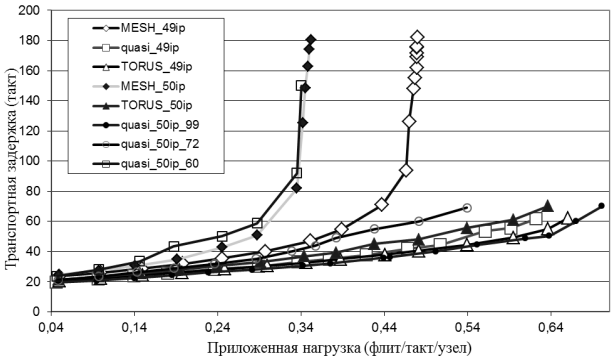


Рисунок 7 – Результаты моделирования топологий для 49 и 50 узлов

Совсем иная ситуация с топологиями для 49 и 50 узлов (рисунок 7). Топология torus для 49 узлов имеет пропускную способность 0,653, а для 50 узлов – 0,585, что на 10,4 % меньше. Для топологии mesh пропускная способность уменьшается еще больше – от 0,405 до 0,301 (на 25,7 %). Однако для количества узлов 99 и 100 ухудшение составило всего 2–3 %. Причиной такой разницы является значительное ухудшение геометрических пропорций топологий для 50 узлов. Для квазиоптимальных топологий данная проблема не актуальна. Кроме того, изменяя приоритет параметров оптимизации для одинакового количества узлов, можно найти топологии с различными характеристиками. Так, для 50 узлов топологии с количеством соединений 60, 72 и 99 имеют пропускную способность 0,289, 0,481 и 0,670 соответственно. То есть, квазиоптимальные топологии предоставляют возможность выбора в зависимости от того, какая характеристика СтнК (пропускная способность или затраты ресурсов) важнее для конкретной задачи: первый вариант топологии имеет пропускную способность, близкую топологии mesh (на 4 % меньше) при уменьшении числа соединений от 85 до 60 (на 29,4 %); второй вариант является примером баланса между производительностью и затратами ресурсов, чего невозможно достичь при использовании топологии mesh и torus; третий вариант квазиоптимальной топологии – близок torus по количеству соединений (на 1 % меньше) и имеет на 14,5 % большую пропускную способность.

Заключение. Предложена быстрая высокоуровневая модель СтнК OCNS на базе сетевой модели OSI с применением языка Java и компонентов Qt, что позволило получить результаты моделирования, близкие к высокоточным HDL-моделям при почти на 2 порядка меньших затратах времени.

Показано, что по результатам моделирования СтнК с 12 узлами с помощью OCNS при одинаковой пропускной способности регулярных и квазиоптимальных сетей расходы ресурсов на соединения в СтнК на основе квазиоптимальных топологических решений являются на 5,9–12,5 % меньшими, чем в сетях с топологиями mesh и torus.

Моделирование СтнК с количеством узлов 49, 50, 99, 100 показало уменьшение пропускной способности СтнК на основе регулярных топологий на 10,4–25,7 % при применении топологий неоптимальной прямоугольной формы. Квазиоптимальные топологии, по сравнению с регулярными топологиями, позволили достичь уменьшения затрат ресурсов на 29,4 % или увеличения пропускной способности СтнК на 14,5 % без потерь по другим параметрам, что свидетельствует об их высокой эффективности и применимости для синтеза СтнК с количеством узлов, достигающим 100.

Библиографический список

1. Fully-synthesizable parameterized NoC implementations library: Netmaker [Электронный ресурс]. – URL: http://www dyn.cl.cam.ac.uk/~rdm34/wiki (дата обращения: 09.04.2015).

2. *Romanov O., Lysenko O.* The Comparative Analysis of the Efficiency of Regular and Pseudo-optimal Topologies of Networks-on-Chip Based on Netmaker // Advances and Challenges in Embedded Computing. Proceedings. – Montenegro, Bar: 2012. P. 13–16.

3. *Романов О.Ю.* Порівняльний аналіз результатів HDL-моделювання квазіоптимальних і регулярних топологій мереж на кристалі // Проблеми інформатизації та управління. Збірник наукових праць. – Київ: НАУ, 2012. – Вип. 3 (39). – С. 124–129.

4. *Genko N., Atienza D., De Micheli G.* A Complete Network-On-Chip Emulation Framework // Design, Automation and Test in Europe, 2005. Proceedings. – 2005. – Vol. 1. – P. 246–251.

5. *Genko N., Atienza D., De Micheli G., Benini L.* Feature-NoC emulation: a tool and design flow for MPSoC // IEEE Circuits and Systems Magazine. – 2007. – Vol. 7. – No. 4. – P. 42–51.

6. *Xiaowen C., Zhonghai L., Jantsch A., Shuming C.* Speedup Analysis of Data-parallel Applications on Multi core NoCs // IEEE 8th International Conference on ASIC, 2009 (ASICON'09). – 2009. – P. 105–108.

7. *Freitas H.C., Navaux P.O.A.* Evaluating On-Chip Interconnection Architectures for Parallel Processing // 11th IEEE International Conference on Computational Science and Engineering Workshops, 2008 (CSE-WORKSHOPS'08). – 2008. – P. 188–193.

8. *Mingsong L., Ying G., Nan G., Qingxu D.* RTNoC: A Simulation Tool for Real-Time Communication Scheduling on Networks-on-Chips // International Conference on Computer Science and Software Engineering. – 2008. – Vol. 4. – P. 102–105.

9. *Whelihan D.* The NOCsim Simulator Users Guide. – [Version 2.0]. – Pittsburgh: CMU, 2003. – 51 p.

10. *Hossain H., Ahmed M., Al-Nayeem A.* GpNoC-sim – A General Purpose Simulator for Network-on-Chip // International Conference on Information and Communication Technology, 2007 (ICICT'07). – 2007. – P. 254–257.

11. *Романов А.Ю., Феськов Д.А.* Разработка программного симулятора сетей на кристалле // Электроника и связь: Электроника и нанотехнологии. – Киев: НТУУ «КПИ», 2011. – Т. 4(63). – С. 48–52.

12. *Concer N.* Design and Performance Evaluation of Network-on-Chip Communication Protocols and Architectures: tesi di dottorato / Concer Nicola. – Padova: Università di Bologna, 2009. – 185 p.

13. *Pandle P.P., Grecu C., Ivanov A., Saleh R.* Design of a Switch for Network on Chip Applications // Proceedings of the 2003 International Symposium on Circuits and Systems, 2003 (ISCAS'03). – 2003. – Vol. 5. – P. V-217–V-220.

14. *Феськов Д.О., Романов О.Ю., Короткий Є.В.* Програмна модель мереж на кристалі з нерегулярними топологіями // Проблеми інформатизації та управління. Збірник наукових праць. – Київ: НАУ, 2013. – Вип. 2 (42). – С. 118–123.

15. *Tran A.N.* On-Chip Network Designs for Many-Core Computational Platforms: Ph.D. thesis / A.N. Tran. – USA, Davis: University of California, 2012. – 156 p.

16. *Starobinski D., Karpovsky M., Zakrevski L.A.* Application of network calculus to general topologies using turn prohibition // IEEE/ACM Transactions on Networking (TON). – 2003. – Vol. 11. – No. 3. – P. 411–421.

17. *Romanov O., Lysenko O.* The Evolutionary Computation Method for the Synthesis of Networks-on-Chip Quasi-optimal Topologies // 2014 IEEE 34th International Scientific Conference on Electronics and Nanotechnology (ELNANO). – Kiev: NTUU "KPI", 2014. – P. 403–407.

УДК. 681.518.3

А.Н. Пылькин, С.В. Филаткин

МОДЕЛИРОВАНИЕ И ОЦЕНКА ХАРАКТЕРИСТИК ПРОТОКОЛА ДЛЯ ПЕРЕДАЧИ ДАННЫХ В РЕАЛЬНОМ ВРЕМЕНИ ПРИ ИСПОЛЬЗОВАНИИ КАНАЛОВ СВЯЗИ С БОЛЬШИМ ВРЕМЕНЕМ ПРИ РАСПРОСТРАНЕНИЯ СИГНАЛА

Ряд задач управления (или контроля) сложных технических комплексов предусматривает получение от объектов контроля оперативной информации об их текущем состоянии. В случаях, когда объекты контроля представляют угрозы безопасности, пункты управления могут находиться от них на удалённом расстоянии, что в условиях отсутствия наземных каналов связи может потребовать использования спутниковых каналов связи, характеризующихся большим временем распространения сигнала.

Современные стандартные протоколы могут неэффективно использовать пропускную способность таких каналов связи, особенно с низкой надёжностью. Решением задач повышения эффективности передачи данных в реальном времени может быть использование специализированных протоколов, ориентированных на применение в каналах связи с большим временем распространения сигнала. Применение описываемого протокола может обеспечить необходимую надёжность для передачи высокочувствительных оперативных данных о состоянии объектов контроля.

Ключевые слова: *распределённые системы, спутниковые каналы связи протоколы передачи данных, каналы связи с большим временем распространения сигнала.*

Введение. При сборе измерительной информации с удаленных измерительных пунктов для передачи данных в реальном времени (репортажные данные) в большинстве случаев используют спутниковые каналы связи, организованные в основном на базе спутников-ретрансляторов на геостационарной орбите.

Для передачи измерительной информации в существующих системах сбора задействуются средства организации компьютерных телекоммуникаций: используются протоколы передачи данных из группы (стека) TCP/IP, а именно протокол передачи данных UDP с присущими ему достоинствами и недостатками, последствием одного из которых могут быть большие потери данных (до нескольких десятков процентов измерений) при передаче в реальном времени по каналам связи с низкой надежностью.

При больших размерах блоков и низкой помехоустойчивости канала связи возможны большие потери данных при передачах в реальном времени, что при измерении быстроменяющихся и контактных (единичных) параметров может привести к невозможности точной оценки состояния, оперативного управления и анализа нештатных ситуаций или отказов объектов контроля.

Цель работы. Теоретическим вопросам повышения эффективности использования каналов связи посвящено множество работ [1-5], где в разной степени отражены как теоретические, так и методологические положения, методы и алгоритмы. Но в основном изучены методы повышения эффективности передачи данных в локальных сетях с использованием протоколов TCP/IP, основанные на оптимизации параметров протоколов TCP/IP, или улучшении различных методов маршрутизации. *Целью* настоящей работы является решение узкоспециализированной задачи более эффективного использования в реальном времени каналов связи с большим временем распространения сигнала – дорогостоящих спутниковых каналов связи.

Теоретические исследования. Для передачи данных в реальном времени предлагается применять протокол со способом повышения вероятности передачи данных в реальном времени с помощью передачи резервных данных.

Отличительные признаки предлагаемого протокола:

1) режимы доставки:

– монопольный: один передатчик – один приемник;

– широковещательный: один передатчик – несколько приемников;

2) условия применения протокола:

– в монопольном режиме: между передатчиком и приемником должен функционировать прямой и обратный каналы связи;

– в широковещательном режиме: должен функционировать широковещательный канал и обратные каналы связи с каждым из приемников;

3) функционирование протокола на передающей стороне:

– к поступающим от средства приема блокам данных добавляют кодовые последовательности, обеспечивающие обнаружение искажений в канале связи, номера блоков, и отправляют в канал связи (первый, высший приоритет выдачи данных в канал связи); отправленные в канал связи пронумерованные блоки заносят в массив отправленных данных;

– поступающие по обратному каналу связи от приемника (или от приемников в широковещательном режиме) запросы на повтор искаженных в канале связи блоков заносятся в массив заявок на повтор;

– в резервное время (признаком которого является отсутствие новых блоков данных, поступающих от средства приема измерительной информации) в канал связи отправляются повторно блоки данных по необслуженным заявкам (второй приоритет выдачи данных в канал связи);

– в резервное время, при условии отсутствия необслуженных заявок на повтор блоков, в канал связи выдаются «резервные» блоки данных – копии блоков, отправленных в канал связи (третий приоритет выдачи данных в канал связи) на приемной стороне в период регистрации;

– принимают основные и резервные (при их наличии) блоки, декодируют для обнаружения искажений; передают пользователю неискаженный блок (в реальном времени); заносят в массив принятых блоков;

– в случае если искажены основной и резервные блоки, формируют запрос на повтор и выдают по обратному каналу на передающую сторону;

– при приеме неискаженного блока данных по запросу на повтор заносят на соответствующее место в массив принятых блоков (заполняют пробел в переданных данных, возникший из-за искажений в канале связи); как вариант (опция) неискаженные блоки по запросу также могут быть переданы пользователю (задержанная передача).

Вероятность потери блока данных в предлагаемом протоколе с передачей резервных блоков данных определяется формулой:

$$P_{птр\ бд} = \left[1-(1-P_o)^{R_{бл}+rzb}\right]^{1+k_{рб}}, \qquad (1)$$

где P_o – вероятность ошибки передачи символа данных в канале связи;

$R_{бл}$ – размер блока данных, бит;

rzb – размер заголовка (размер служебных данных блока), бит;

$k_{рб}$ – число передаваемых резервных блоков данных.

Передача по отдельным измеряемым параметрам дополнительных резервных блоков данных позволяет повысить вероятность полноты доставки измерительной информации приемникам в реальном времени.

На рисунках 1 – 3 приведены диаграммы изменения вероятности потерь блоков данных в зависимости от количества передаваемых резервных блоков.

Без передачи резервных блоков (рисунок 1) вероятность потерь блоков данных остается высокой в широком диапазоне помехоустойчивости канала связи – в диапазоне $L_{oss} = 10^{-6}...10^{-3}$.

С передачей одного резервного блока (рисунок 2) зона высокой вероятности потерь блоков сокращается: $L_{oss} = 10^{-4}...10^{-3}$.

С передачей двух резервных блоков (рисунок 3) высокая вероятность потерь блоков возможна только при низкой помехоустойчивости канала связи: $L_{oss} = 10^{-3}$.

Экспериментальные исследования. Для обеспечения передачи данных в реальном времени пропускная способность канала связи должна превышать среднюю интенсивность поступления измерительной информации в реальном времени.

Указанное превышение может быть достаточным для дополнительных передач данных по запросам обратной связи, формируемым приемником измерительной информации. При наличии временного резерва такие запросы передатчиком измерительной информации обрабатываются со вторым приоритетом и компенсируют искажения и потери данных в канале связи в темпе передачи измерительной информации. К концу периода регистрации практически все заявки на повторную передачу будут выполнены и на приемной стороне будет сформирован массив потока измерительной информации с гарантированной полнотой (все данные будут переданы приемнику).

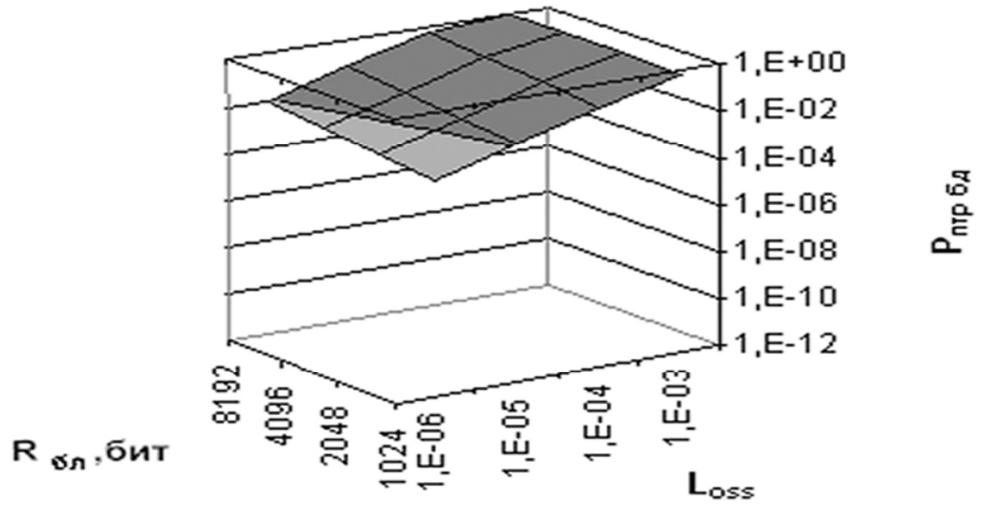


Рисунок 1 – Вероятность потери блока данных $P_{птр\ бл}$ (размером $R_{бл}$) в зависимости от вероятности искажения символа L_{oss} в канале связи для протокола без передачи резервных блоков

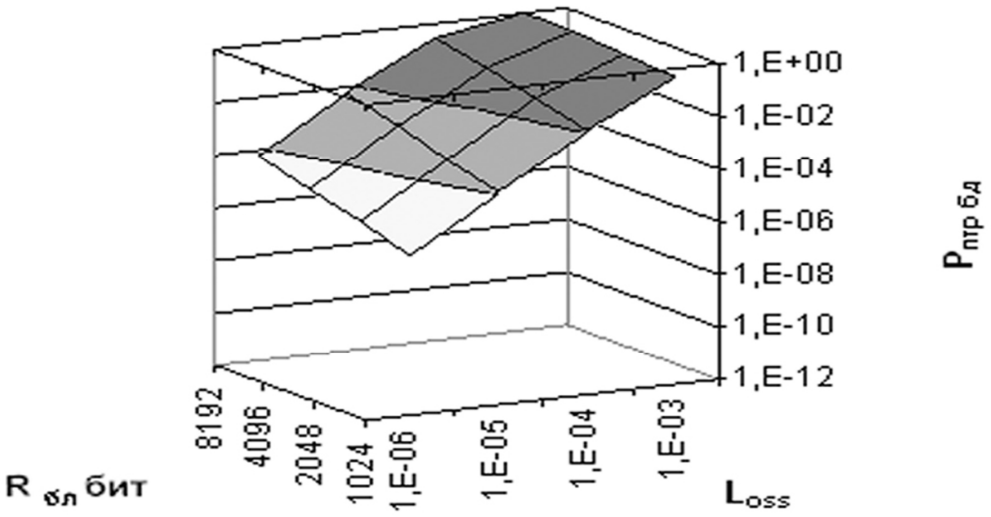


Рисунок 2 – Вероятность потери блока данных $P_{пр\,бл}$ (размером $R_{бл}$) в зависимости от вероятности искажения символа L_{oss} в канале связи для протокола с передачей одного резервного блока

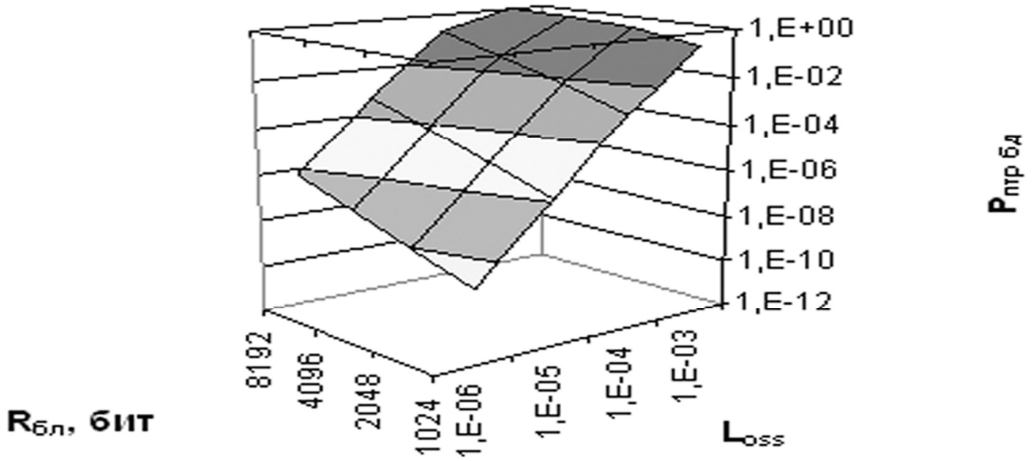


Рисунок 3 – Вероятность потери блока данных $P_{пр\,бл}$ (размером $R_{бл}$) в зависимости от вероятности искажения символа L_{oss} в канале связи для протокола с передачей двух резервных блоков

Для всех блоков, которые не были искажены при первой передаче, будет гарантировано время доставки.

Необходимый для этого коэффициент превышения можно оценить статистическим коэффициентом $\overline{k_{пвт}}$ повторов передач блоков данных:

$$\begin{aligned}\overline{k_{пвт}} &= P_{прд\,бд} + 2P_{пр\,бд}P_{прд\,бд} + 3P_{пр\,бд}^2P_{прд\,бд} \\ &+ 4P_{пр\,бд}^3P_{прд\,бд} + \ldots = \left(1 - P_{пр\,бд}\right) \\ &+ 2P_{пр\,бд}\left(1 - P_{пр\,бд}\right) + 3P_{пр\,бд}^2\left(1 - P_{пр\,бд}\right) \\ &+ 4P_{пр\,бд}^3\left(1 - P_{пр\,бд}\right) + \ldots = 1 + P_{пр\,бд} \\ &+ P_{пр\,бд}^2 + P_{пр\,бд}^3 + P_{пр\,бд}^4 + \ldots = 1/\left(1 - P_{пр\,бд}\right),\end{aligned} \quad (2)$$

где $P_{прд\,бд}$ – вероятность правильной передачи блока данных, определяемой соответственно:

$$P_{прд\,бд} = \left(1 - P_o\right)^{R_{бл}+rzb}. \quad (3)$$

Значение коэффициента зависит от помехоустойчивости канала связи (L_{oss}) и от размера блоков $R_{бл}$, передаваемых в канал связи (с учетом размера служебных данных rzb, передаваемых по каналу связи).

Изменения $\overline{k_{пвт}}$ при этом приведены в таблице и на рисунке 4.

При достаточно высокой помехоустойчивости канала связи ($L_{oss} = 10^{-6}...10^{-5}$) достаточно, чтобы его пропускная способность превышала интенсивность поступления измерительной ин-

формации на несколько процентов (при удовлетворительной вероятности передачи блоков без искажений).

Чем ниже реальная помехоустойчивость канала связи, тем выше должно быть превышение пропускной способности канала связи над интенсивностью поступления измерительной информации.

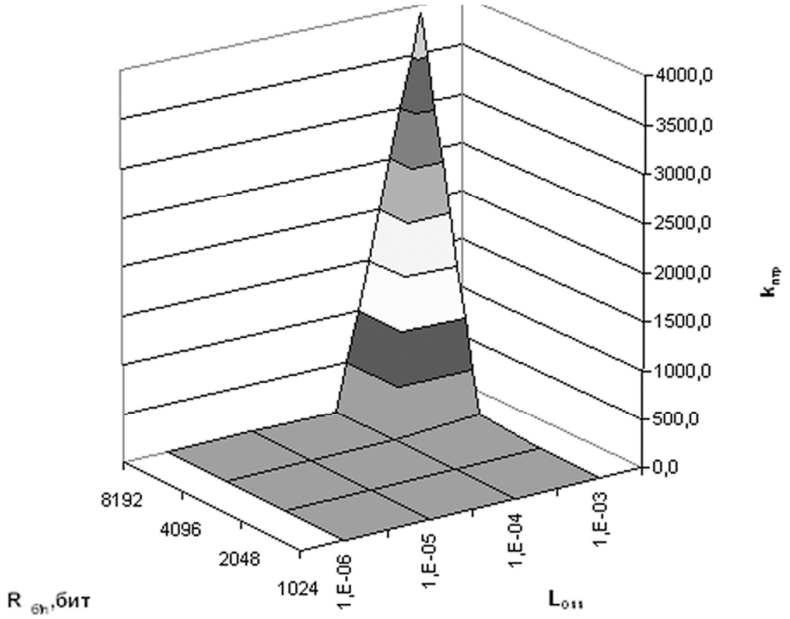


Рисунок 4 – Необходимое превышение пропускной способности канала связи над интенсивностью поступления измерительной информации (соответствует $\overline{k_{пвт}}$) для гарантии передачи данных в реальном времени

Необходимое превышение пропускной способности канала связи над интенсивностью поступления измерительной информации (соответствует $\overline{k_{пвт}}$) для гарантии передачи данных в реальном времени

$R_{бл}$, бит	L_{oss}			
	10^{-6}	10^{-5}	10^{-4}	10^{-3}
1024	1,001105	1,011101	1,116731	3,017873
2048	1,00213	1,021508	1,23715	8,407
4096	1,004185	1,042644	1,518345	65,24108
8192	1,008306	1,086238	2,287001	3928,994

Практически недопустимой является низкая помехоустойчивость канала связи (выброс вверх на диаграмме). Передача блоков данных практически невозможна.

Выводы. Предлагаемые решения по сравнению с известным способом (передача измерительной информации в реальном времени по протоколу UDP) позволяют на несколько порядков уменьшить вероятность потери (искажений) передаваемых измерений с гарантией времени их доставки.

При предлагаемой реализации протокола передач данных по спутниковому каналу связи эффективная пропускная способность:

– приближается к максимальной пропускной способности канала связи;

– практически не зависит от выбора параметров управления протоколом (более точно: «протокол обладает более широким диапазоном параметров управления, в котором эффективно используются возможности канала связи»);

– оптимальна (всегда обеспечивается максимально возможное использование пропускной способности канала связи) при колебаниях реальной помехоустойчивости (при изменении P_o в широком диапазоне) передач по спутниковому каналу связи.

К некритичным недостаткам предлагаемых протоколов можно отнести:

– увеличение ресурсов оперативной памяти, задействуемых на приемной стороне для реализации протокола;

– использование протокола требует соответствующих средств и на приемной стороне, и на передающей стороне.

Библиографический список

1. Telecommunication standardization sector of International Telecommunication Union. Y/1541, 12.2011. Series Y: Global information infrastructure, Internet protocol aspects and next-generation networks. Internet protocol aspects – Quality of service and network performance. Net-work performance objectives for-based services.

2. *Корячко В.П., Перепелкин Д.А., Перепелкин А.И.* Повышение эффективности функционирования корпоративных сетей при динамических изменениях в их структуре и нагрузках на линиях связи // Вестник Рязанского государственного радиотехнического университета. 2010. № 33. С. 49-55.

3. *Корячко В.П., Перепелкин Д.А., Иванчикова М.А.* Алгоритм адаптивной маршрутизации в корпоративных сетях нескольких провайдеров связи // Вестник Рязанского государственного радиотехнического университета. 2013. № 2 (44). С. 52-56.

4. *Корячко В.П., Перепелкин Д.А., Иванчикова М.А.* Интеграция распределённых программных приложений на основе маршрутизации по содержимому сообщений // Вестник Рязанского государственного радиотехнического университета. 2014. № 47. С. 75-83.

5. *Корячко В.П., Лукьянов О.В., Шибанов А.П.* Нахождение скрытого параллелизма протоколов для улучшения характеристик сети передачи данных полигонного измерительного комплекса // Вестник Рязанского государственного радиотехнического университета. 2014. № 47. С. 68-75.

УДК 007:681.512.2

И.Ю. Каширин, О.И. Каширина

ОБЗОР КОНЦЕПЦИЙ ФОРМАЛЬНОГО ИССЛЕДОВАНИЯ ИНСТРУМЕНТАЛЬНЫХ ПРОГРАММНЫХ СРЕДСТВ

Рассматриваются методы формализации программного обеспечения, их достоинства и недостатки. В заключении определяются современные проблемы математического анализа инструментальных программных средств.

Ключевые слова: *анализ программ, алгоритмические алгебры, инструментальные программные средства.*

Введение. Развитие математических теорий в области программирования берет начало с известных работ Черча, Геделя, Тьюринга, Поста, Клини [1] в 30-х – 40-х годах XX века. Наиболее известными результатами стали научные работы 1936 г. [2,3], предопределившие появление средств вычислительной техники и ее математического обеспечения. На основе математически точного конструктивного определения понятия частично-рекурсивной функции, тождественного понятию любой вычисляемой с помощью какого-либо алгоритма функции, удалось существенно уточнить важнейшее математическое понятие – понятие алгоритма. Методология конструктивного доказательства тождественности, использованная при этом, была многократно использована позднее в других теоретических концепциях, направленных на формальный анализ алгоритмов.

Теория программирования, интенсивно развивающаяся с 1950-х годов, заимствовавшая у теории алгоритмов само понятие программы, тем не менее весьма редко использовала теоретико-алгоритмические методы. Она была направлена в основном на получение языковых, инструментальных и методологических средств, улучшающих характеристики трудоемкости проектирования и функциональной надежности программных систем.

Работы В.М. Глушкова, А.П. Ершова, А.А. Ляпунова, Ю.И. Янова [4] в области программирующих программ и алгоритмических алгебр привели к появлению концепции “башни языков” [5], в которой удачно отражены принципы поступательного формирования новых инструментальных программных средств на основе анализа ранее имевшегося программного обеспечения.

Согласно этой концепции анализ имеющихся инструментальных средств позволяет выявлять их слабые и сильные стороны, создавать на основе такого анализа новые, более мощные (алгоритмически сложные) языковые конструкции, которые рано или поздно составляют основу нового инструментария.

Одно из положений концепции утверждает необходимость проектирования перспективных программных средств в рамках самого мощного на текущий момент времени инструментария. По этим принципам сформировалась известная цепочка: “объектный код – программирующие программы – язык Ассемблера – Фортран – Алгол – Паскаль/Си – Си# – case-ориентированные инструментальные среды и интеллектуальные средства проектирования” [6].

В 1960-х – 1970-х годах появляется понимание информационной направленности средств вычислительной техники [7], в корне меняющей

представление об ЭВМ как о средстве для выполнения математических вычислений. Вместе с этим пониманием выявляется настоятельно требующая эффективного решения *проблема сложности программного обеспечения* [8]. На основе системного подхода формируется классификация языков программирования и инструментальных программных средств. Выделяются универсальные и проблемно-ориентированные языки, функциональные и логические системы, автоматизированные системы с искусственным интеллектом и базами данных [9, 10, 11].

1980-е – 1990-е годы характеризуются полностью сформулированными основными проблемами теории программирования, к которым можно отнести:

- исследование надежности программ [12],
- автоматизацию проектирования программных систем [13],
- исследование корректности автоматизированных систем, в том числе машинными методами [14],
- разработку средств структурирования и оптимизации программного обеспечения.

Все перечисленные проблемы весьма существенно взаимосвязаны и не могут быть решены без должной математической формализации. Современные научные исследования в области теории программирования используют средства и методы следующих математических теорий:

- универсальные (абстрактные) алгебры,
- теория графов,
- формальные аксиоматические системы и теория доказательств,
- теория формальных грамматик и языков,
- теория множеств,
- теория унификации.

Чаще всего при исследовании программных систем применяются сразу несколько математических формализмов, один из которых является доминирующим.

Цель работы – обзорный анализ известных концепций формального анализа программного инструментария с выявлением соответствующих классов задач, в которых эти формализмы наиболее эффективны для решения актуальных задач.

1. Аксиоматика для формального анализа программных систем.

Под *инструментальными программными средствами* понимается класс программ, предназначенных для автоматизации процесса проектирования программных систем. К таким инструментальным средствам можно отнести:

- языки программирования,
- библиотеки программ,
- библиотеки классов объектно-ориентированных программ,
- многооконные программные среды со встроенными отладчиками и компиляторами,
- протоколы сетей ЭВМ.

При дальнейшем изложении понятие программного инструментария будет ограничено рамками инструментальных средств, имеющих языковую основу, т.е. например, алгоритмические языки, библиотеки классов, сервисные уровни протоколов сетей [9].

Уровень инструментария, в рамках которого проектируются программы, определяет сложность программных систем, которые можно в нем реализовать с приемлемой трудоемкостью. Более мощный инструментарий дает возможность создавать более сложные программы. Этот факт определяет постоянное развитие систем, поскольку более сложные инструментальные средства проектируются на основе более простых. Самым простейшим программным средством является объектный код, который проще всего представить в рамках одного из основных формализмов теории алгоритмов – машины Тьюринга.

Для новых языков программирования (ЯП) и инструментальных программных систем используются более сложные математические формализмы для исследования алгоритмов, составленных в рамках этих языков и инструментальных систем.

Так, например, в современных исследованиях преобладают вопросы оценки концептуальной целостности, целесообразности или избыточности тех или иных синтаксических конструкций ЯП, а также места ЯП в системе других известных универсальных и проблемно ориентированных языков на основе построения единой системы интерпретации базовых конструкций, определяющих свойства этих языков.

Разработка нового алгоритмического языка обычно связана с появлением теоретического и программного инструментария, успешно решающего некоторый класс задач в определенной предметной области, которая может принадлежать как к прикладным, так и к фундаментальным областям. Этот класс задач, как правило, был ранее или не определен, или же считался достаточно сложным для использования известных методов решения.

Вместе с тем, новый инструментарий только тогда становится основой нового языкового средства, когда при сколь угодно сложной внутренней реализации он имеет достаточно простую, доступную программисту, внешнюю модель данных со строго классифицированной системой операций.

Степень соответствия языкового инструментария предметной области задач, для решения которых он предназначен, в большей части определяется строгостью отображения всех информативных элементов предметной области в элементы модели данных и операционный базис языковых средств.

Для обеспечения строгости такого отображения и пригодности инструментария к реализации хорошо обозримых и верифицируемых программ необходимо формировать основные конструкции языковых средств, достаточно ясно представляя себе их структурные, функциональные и композиционные свойства. Анализ этих свойств дает возможность отнести языковую инструментальную систему к определенному классу систем, в которых может производиться многокритериальное сравнение языковых конструкций в рамках как одного языка или семейства языков, так и однотипных конструкций языков разных семейств.

Разнообразие классов ЯП, которые в общем случае подчиняются разным основаниям классификации, определяется пространством их основных характеристик.

В то же время многообразие инструментальных средств дает возможность выбора конкретного инструмента для проектирования сложной программы определенной направленности. К *сложным* программным системам принято относить системы, обладающие следующими признаками :

- стратифицированный характер системы,
- нижний уровень логической структуры программы не может быть определен с достаточной точностью,
- наличие сложных внутриэлементных и межэлементных связей,
- построение различных элементов системы по нескольким различающимся методам,
- наличие в системе более простой первоначальной основы.

Поясним эти признаки. Стратифицированный характер системы представляет собой такое ее разбиение на элементы, при котором образуется некоторая иерархия подсистем. При этом одни подсистемы используют результаты работы других, те, в свою очередь, третьих и т.д.

Невозможность определения точного нижнего уровня логической структуры говорит о том, что различные проектировщики могли бы поразному выбрать этот уровень, поскольку сложность задачи не позволяет найти критерии его точной идентификации.

Сложность элементных связей в системе предполагает, что внутриэлементные связи оказываются сильнее межэлементных. Это приводит к выделению в системе некоторых локальных модулей разных уровней.

Использование различных методов для проектирования различных элементов системы свидетельствует о сложности задачи, для решения которой она предназначена.

Наличие первоначальной основы предполагает эволюционное проектирование системы от простого к сложному, при котором можно выделить простые подсистемы, способные функционировать самостоятельно.

Перечисленные признаки свидетельствуют о необходимости формального исследования таких программных систем на основе анализа соответствующих инструментальных программных средств.

Рассмотрим кратко результаты, достигнутые до настоящего времени в области формального анализа программных систем инструментальной направленности, т.е. систем, предназначенных для реализации других программ.

Наиболее популярным математическим аппаратом, используемым при анализе программ, является алгоритмическая алгебра. Ее широкое использование для решения практически всех проблем теории программирования обусловлено выразительными возможностями и хорошей исследованностью аппарата универсальных алгебр [15–17].

На основе системы алгебраических операций композиции *схемных записей алгоритмов* Ю.И.Яновым был спроектирован оригинальный формализм. Он позволял на основе алфавита из пропозициональных символов операторов A_1, A_2, …, A_n и предикатов p_1, p_2, .., p_m, а также символов левой и правой полускобок $\lfloor$, $\rfloor$ записывать схемы алгоритмов, которые можно было использовать для описания интерпретации разнообразных программных конструкций. В этом формализме строка

$$A_i \; p \lfloor \; A_{i+1}, \ldots, A_{i+n} \rfloor \; A_k$$

означала, что после выполнения оператора A_i в случае истинности предиката p выполняется последовательность операторов A_{i+1}, …, A_{i+n} , в противном случае переход осуществляется к оператору A_k. Схемные записи могли использовать операции математической логики [18] для построения более сложных предикатов и операции композиции записей. Янов получил эффективный критерий, позволяющий для любой пары схемных записей установить их равносильность или частичную равносильность. На основе схемных записей им была построена логическая система тождественных преобразований, полная в том смысле, что для конкретной схемной записи

любая, эквивалентная ей, запись единственна, т.е. может быть получена из нее только с помощью специально оговоренной системы преобразований.

Несколько отличающиеся от алгоритмических схем Янова, Ершовым и Ляпуновым [19] были разработаны другие алгебраически-представимые формализмы, в рамках которых исследовалось *понятие эквивалентности алгоритмов*. Эквивалентность разбивалась на три основные типа:

- функциональная эквивалентность всех результатов сравниваемых программ,
- операционная эквивалентность по истории выполнения программ,
- формальная эквивалентность по множеству всех цепочек операторов, порождаемых схемами алгоритмов.

Эти типы эквивалентности в настоящее время широко используются при построении логических семантик программ. Впоследствии была доказана теорема о равносильности всех трех типов эквивалентности, что сделало возможным использование единого предикатного символа для обозначения любого из типов эквивалентности, и при необходимости преобразовывать эти типы друг в друга [20].

Весьма строгое определение понятия алгоритмической алгебры было дано в [21]. Этот вид алгебр представляет собой двуосновную алгебру с множествами-носителями, составленными из программных примитивов. Так, например, может быть определено два основных множества: множество условий и множество действий. Сигнатура алгебры содержит операции композиции, позволяющие образовать линейные и разветвляющиеся схемы алгоритмов. Множество-носитель алгебры содержит операторы “пропустить” и “отказать”, аналогичные алгебраическим единицам и нулям универсальных алгебр и характерные для многих других алгоритмических алгебр [22]. Микропрограммные алгебры имеют свойства ассоциативности операции композиции, дистрибутивности операции линейной композиции относительно операции логического разветвления. Исследование алгоритмических алгебр привело к появлению нового понятия – *регулярных алгоритмов* или алгоритмических схем программ, представляющих собой такие алгоритмы, в которых нет пересекающихся или потенциально бесконечных циклических конструкций, а также не содержащих переходов неструктурного типа [20]. В рамках регулярных схем микропрограммных алгебр представимы логические схемы алгоритмов Янова.

Более сложными концепциями, основанными на алгоритмических алгебрах, являются семантические алгебры [23], позволяющие рассматривать не только эквивалентные преобразования программ, но и семантику инструментальных языковых программных средств, а также допускающие верификацию. Компьютерные алгебры весьма успешно применяются в настоящее время и для исследования довольно сложных автоматизированных систем.

Попытки получения унифицированного алгебраического формализма, который можно было бы с одинаковым успехом использовать для анализа разнообразных классов программ, привели к созданию алгебры процессов [24], в рамках которой допустимо описывать параллельные вычисления и даже использовать при этом понятия каналов передачи данных. Алгебра процессов использует понятие сценариев процессов в некоторой вычислительной системе, задаваемое как в форме алгебраического кортежа, так и в графическом виде (рисунок 1) с использованием сетей Петри [25]. На рисунке в прямоугольниках располагаются сценарии отдельных процессов. В круглых скобках в сети указываются контексты событий или сообщений и их имена.

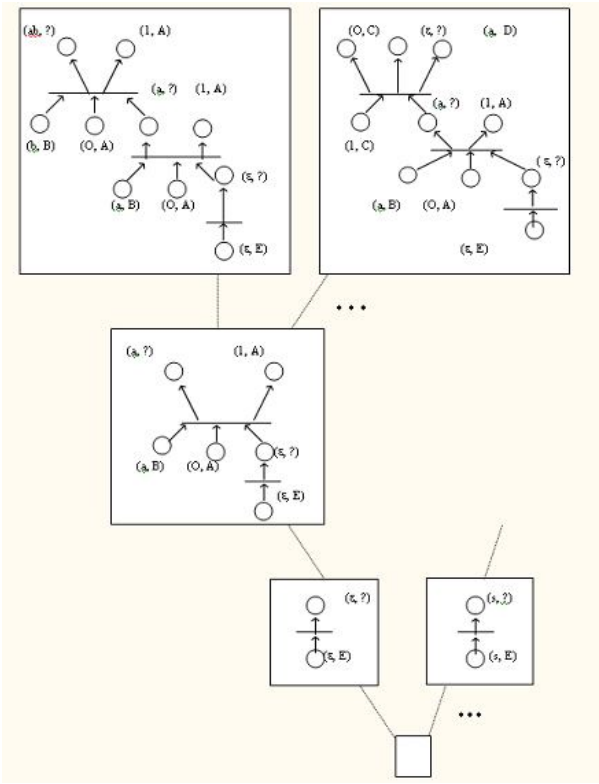


Рисунок 1 – Графическое представление программы в алгебре процессов

Набор сценариев описывается множеством алгебраических выражений, определяющих, какие события, в какой последовательности и в зависимости от каких условий изменяют свои состояния. Алгебра позволяет рассматривать

конфликтующие события с принятием решения о снятии конфликта в пользу одного из событий.

Практически каждая из алгоритмических алгебр, используемых при исследовании программных систем, является областью интерпретации соответствующей формальной аксиоматической системы [26].

Одной из наиболее известных аксиоматических систем, созданных для анализа программ и положенных в основу многих других аксиоматик, названных *программными логиками*, является логическая система Хоара [27]. Эта система замечательна введением нестандартного синтаксиса выражений формально-логической системы, позволившего не только исследовать схемы программ, но и описывать семантику языков программирования.

Основная идея аксиоматического подхода к формализации семантики состоит в том, чтобы свойства оператора языка выразить как отношения между программными переменными, представленными в виде логических формул. В этом случае явно описываются свойства текущего состояния программы.

Аксиоматическая семантика языка программирования (ЯП) задается фиксацией некоторого формального языка утверждений о свойствах программ. Выводимые в этой аксиоматической системе утверждения характеризуют свойства программ. В аксиоматической системе Хоара язык утверждений о свойствах программ содержит конструкции вида {P}A{Q}, называемые тройками Хоара. Тройки представляют собой утверждения о частичной корректности программ. При этом предикаты P и Q представляют собой соответственно пред- и постусловия программы A.

Свойства троек Хоара устанавливают возможность преобразования операторов программы с сохранением неизменности предикатов или, наоборот, преобразования предикатов P и Q при неизменности программы. Некоторые аксиомы и правила вывода для алголоподобного языка приведены в таблице 1.

Аксиоматическая система Хоара была модифицирована во многих научных работах в целях улучшения ее функциональных свойств [27].

Существенным уточнением предикатов предусловия и постусловия явился формализм, предложенный Дейкстрой [5], основанный на понятии *слабейшего предусловия.* Это понятие определяется как самое слабое условие, которому достаточно подчинить начальное состояние для того, чтобы выполнение программы завершилось и дало состояние, удовлетворяющее постусловию программы. Вместе с тем, семантики ЯП, основанные на дейкстровской системе, обладали рядом весьма существенных недостатков, рассмотренных в [27]. Там же приводится более точное понимание предикатов слабейшего предусловия wp для программы P через сопряженное преобразование p областей данных υ и постусловия ψ. Рассматривается также возможность использования предиката сильнейшего постусловия sp. Если обозначить некоторые программные фу4нкции через f и φ, а их аргументы (константы и переменные) большими и малыми латинскими буквами, то, используя понятие сопряженного преобразования, можно убедиться в справедливости следующих выражений [5]:

$wp (P, \psi) (\upsilon) = \psi (p (\upsilon))$,

$wp (P, \psi) = p^* (\psi)$, $p^* (\psi) (\upsilon) = \psi (p (\upsilon))$,

$sp(x:=f(a,b,...,x),\varphi(a,b,...,x)) = \exists x'(x:=f(a,b,...,x')\wedge \varphi(a,b,...,x'))$,

$sp(P, \neg\varphi) = \neg sp(P, \varphi)$, $sp(P, \varphi_1\wedge\varphi_2)=sp(P, \varphi_1) \wedge sp(P, \varphi_2)$,

$sp(P, \varphi_1\vee\varphi_2)=sp(P, \varphi_1) \vee sp(P, \varphi_2)$.

Семантика языковых операторов в этом случае может быть уточнена, например, для оператора if β then R else S fi; справедливо

if β then wp(R,g) else wp(S,g) fi.

Формализмы, основанные на понятии слабейшего предусловия, в целом являясь математически строгими, обладают весьма существенным недостатком: они являются довольно сложными для их использования проектировщиком сложных программных систем непосредственно в процессе проектирования инструментария. Это обстоятельство побудило исследователей к созданию более простых, хотя и менее универсальных средств, дающих возможность анализировать эквивалентные программные трансформации.

Одним из таких средств является предикативное программирование [28], основанное на описании спецификации программ средствами логических предикатов. Так, например, если обозначить большими латинскими буквами логические спецификации, малыми буквами – условные выражения, выделенной буквой v – вектор начальных значений переменных, а знаком ";" – операцию композиции, то можно записать следующие теоремы предикативного программирования.

$P ; (Q ; R) => (P ; Q) ; R$,

$(\forall v.P) \vee (\exists! v.P)) => P ; (Q ; R) = (P ; Q) ; R$,

(if b then P else Q) = (if $\neg$ b then Q else P),

((if b then P else Q) ; R = (if b then (P ; R) else (Q ; R)),

(P or Q) or R = P or (Q or R),
P or P = P,
P or Q = Q or P,
P ; (Q or R) = (P ; Q) or (P ; R),
((∀v.P) ∨ (∀v.Q) ∨ (¬∀v.P∨Q)) =>
((P or Q) ; R) = P
or (if b then Q else R),
(if b then (P or Q) else (P or R) = P or (if b then Q else R),
где знак "=>" обозначает выводимость одного выражения из другого, а ∀, ∃! – кванторы всеобщности и существования единственного элемента.

2. Формальный анализ программ, основанный на алгоритмических алгебрах.

Другим эффективным формализмом для исследования программ является математический аппарат, ориентированный на абстрактно-алгебраические структуры [29], использующий предикатно-функциональные элементы для доказательства функциональной эквивалентности алгоритмов.

Аналогичные алгебраико-логические модели были построены для различных классов программных систем, например, для программ, использующих рекурсивные процедуры [30]. Следуя основным понятиям этих моделей, две процедуры называются эквивалентными, если для любой заданной входной информации они или обе останавливаются и выдают одинаковый результат, или обе не завершаются. Программы при этом рассматриваются как рекурсивные схемы с интерпретацией.

Язык рекурсивных схем содержит операции присваивания, последовательной композиции, ветвления и рекурсии.

В подходе де Баккера процедура интерпретируется как минимальная неподвижная точка трансформации, определяемая телом процедуры.

Логика на основе модифицированной алгебры процессов [31] была разработана для анализа программных систем с возможностью учета наиболее полного набора свойств программ, рассматриваемых как множество процессов. Важным результатом при этом является использование не только предыдущих результатов в области представления процессов, но и основные понятия хоаровской логики [20]. Логика располагает специальным языком, в котором приняты следующие обозначения: action(*a*, s) – обозначает, что процесс будет рассматриваться, если выполняется *a* в начальном состоянии s, effect (s, *a*) означает результирующее состояние выражения. Если действие успешно выполняется, то это формально записывается так:

$$(a,\ s) \xrightarrow{\text{action(s)}} (\surd, effect(s,a)),$$

где (a, s) представляет процесс, а в начальном состоянии s и effect (s, *a*) называется заключительным состоянием.

Правила вывода этой формальной системы приведены в таблице 2, где через *a* обозначено любое действие из некоторого множества действий A. Выделенный символ δ обозначает тупиковое действие, а выражение $< x \mid E >$ – x есть решение выражения E. Важным является то, что эта формальная система обладает полнотой, что делает ее применимой в системах автоматизированного преобразования программ. Запись {P} A {S} означает, что при наложении условий на входные данные (предусловие) после выполнения оператора A будет выполняться постусловие S. "->" – логическая импликация, "⇒" – символ выводимости.

Множество научных трудов посвящено детерминированной [32] и квазидетерминированной [33] логикам процессов и программ, для которых были получены результаты по преобразованию программ к регулярным схемам, их эквивалентная трансформация и оптимизация.

Существуют модификации алгебраико-логических формализмов исследования программ, отличающиеся от других более хорошими свойствами. Это касается программной реализации на их основе алгоритмов оптимизации и автоматического построения программ [34] .

Современные достижения в области математической логики и абстрактной алгебры позволили создать математические системы и рассматривать некоторые особенности схем алгоритмов, связанные с процессами их создания и последующего преобразования, среди таких систем, в частности, можно упомянуть:

- исследование так называемых "неизбежных" свойств программ [36],
- анализ фрагментов программ, касающихся доступа к наборам данных древовидной и сетевой структур, а также прямого доступа по ключам [37],
- исследование программ, использующих теоретическую концепцию нейросетей [38],
- анализ параллельных программ [39].

Оригинальные решения в области логических семантик для программных логик привели к возможности выделения их более точных классов, имеющих свойства полноты. К таким логическим системам относятся:

- системы с семантикой путей (трасс) вычислений [40],
- денотационная семантика [41],
- унификационная семантика[42].

- исследование свойств программ с помощью трехзначной логики [35],

Разнообразие алгебраических и логических подходов к формальному анализу программ свидетельствует о разнообразии и сложности проблем, существующих в этой области. Главными из них, как следует из ранее изложенного, являются:

- создание удобных формализмов для исследования программ широкого класса (от императивных до параллельных),

- создание наиболее точных семантик или интерпретаций логических выражений в программных логиках.

Таблица 1 – Аксиомы и правила вывода аксиоматической системы для алголоподобного языка

Обозначение	*Содержание*	*Примечание*
HA.NL	{ Q } null { Q }	Пустой оператор
HA.ASG	{Q (x←e)}x:=e {Q}	Простое присваивание
HA.ASGR	{Q(R}←upd (R, a, e))} R.a:=e {Q}	Присваивание записи
HR.SEQ	{P} A_1 {S}, {S} A_2 {Q}⇒ {P} A_1; A_2 {Q}	Последовательность операторов
HR.GR	{P_{i-1}}A_i {P_i} для i = 1,...,n ⇒ {P_0}A_1 ;A_2;...;A_i ; ...;A_n {P_n}	Групповой оператор
HR.IF	{P&α}A_1{Q},{P&~α}{Q} ⇒ {P} if α then A_1 else A_2 {Q}	Условный оператор
HR.WHL	{ inv & α }A {inv} ⇒ {inv} while α do A {¬α & inv}	Оператор цикла while-do
HR.PRED	P -> Q, {S} A {Q} ⇒ { P } A { Q }	Усиление предусловия
HR.POST	S -> Q, {P} A {S} ⇒ { P } A { Q }	Ослабление постусловия

- упрощение существующих формальных средств анализа схем алгоритмов средствами программных спецификаций [15],

- получение формальных и инструментальных средств, в рамках которых возможно автоматизированное исследование программ,

- разработка формализмов практической направленности (удобных для оптимизации, верификации, автоматического проектирования и структурирования программных систем),

- создание математически точных средств оценки структурности [18] и концептуальной целостности [43] инструментальных программных средств.

Существенными недостатками рассмотренных теоретических разработок являются:

- сложность математических формализмов, требующих от программиста знания многих математических аппаратов,

- возможность внесения ошибок на концептуально-логическом уровне проектирования,

- высокая трудоемкость применения методов повышения надежности,

- неполнота и неразрешимость многих формальных систем, используемых в качестве программных логик,

- узкая направленность удобных и простых формальных средств на конкретный тип.

Каждый из существующих формальных аппаратов решает только некоторую часть из перечисленных проблем и лишь косвенно способствует решению остальных задач.

Таблица 2 – Правила вывода этой формальной системы

a ∈ A:	$(a,\ s) \xrightarrow{a(s)} (\surd, s(a))$	$(\text{if } a(s) \neq \delta)$
+:	$\dfrac{(x,s) \xrightarrow{a} (x',s')}{(x+y,s) \xrightarrow{a} (x',s')}$	$\dfrac{(x,s) \xrightarrow{a} (\surd,s')}{(x+y,s) \xrightarrow{a} (\surd,s')}$
	$\dfrac{(y,s) \xrightarrow{a} (y',s')}{(x+y,s) \xrightarrow{a} (y',s')}$	$\dfrac{(y,s) \xrightarrow{a} (\surd,s')}{(x+y,s) \xrightarrow{a} (\surd,s')}$
.:	$\dfrac{(x,s) \xrightarrow{a} (x',s')}{(xy,s) \xrightarrow{a} (x'y,s')}$	$\dfrac{(x,s) \xrightarrow{a} (\surd,s')}{(xy,s) \xrightarrow{a} (y,s')}$
recursion:	$\dfrac{(\langle t_x \mid E\rangle,s) \xrightarrow{a} (y,s')}{(\langle x \mid E\rangle,s) \xrightarrow{a} (y,s')}$	$\dfrac{(\langle t_x \mid E\rangle) \xrightarrow{a} (\surd,s')}{(\langle x \mid E\rangle,s) \xrightarrow{a} (\surd,s')}$
π_n:	$\dfrac{(x,s) \xrightarrow{a} (x',s')}{(\pi_{n+1}(x),s) \xrightarrow{a} (\pi_n(x'),s')}$	$\dfrac{(x,s) \xrightarrow{a} (,s')}{(\pi_{n+1}(x),s) \xrightarrow{a} (,s')}$

Особенно сложной проблемой является получение простого средства для исследования инструментальных систем, допускающего в то же время применение в рамках автоматизированных средств анализа. Сложность имеющихся формализмов не позволяет их реально применить в доказательствах правильности и оптимизации сложных программных систем.

Перспективным механизмом решения рассмотренных проблем является применение относительно недавно сформировавшейся теории унификации, весьма быстро определившей свою сферу задач, систему понятий и области практического применения [44].

В то же время методы исследования алгебраических термов, используемые в теории унификации, вполне пригодны для термов алгоритмических алгебр и программных логик [45]. Положительным свойством использования методов унификации является их исследованность и пригодность для выполнения автоматических преобразований над термами [46].

Этими соображениями определяется актуальность разработки и исследования наиболее универсальных с точки зрения теории программирования формализмов, использующих методологию теории унификации.

3. Современные задачи формального анализа программного инструментария.

Учитывая все ранее сказанное, а также проблемы, существующие в области формального исследования инструментальных программных систем, можно сформулировать следующие задачи, решение которых позволит существенно эффективнее использовать существующие научные результаты для практических целей программирования. К ним относятся:

- разработка простых и эффективных средств оптимизации программного инструментария, пригодных одновременно для автоматизированного проектирования программных систем на основе теории унификации,
- получение в рамках единого формализма методов исследования корректности (доказательства правильности программы) и структурирования программы для простого внесения изменений,
- упрощение инструментария программирования на основе case-подхода с целью повышения надежности программ и сокращения трудо-

емкости их разработки,

- определение критериев уровня концептуальной целостности инструментальных программных средств,

- построение языка для удобной спецификации программ,

- выработка рекомендаций по автоматизации исследования корректности программ на основе ранее построенных формализмов,

- определение признаков структурного проектирования объектно-ориентированных программ, допускающих использование методов унификации,

- разработка эффективных проектных решений, ориентированных на теоретико-унификационные методы.

Заключение. Научные результаты, полученные в рамках многочисленных подходов к формализации программного инструментария, а также задачи, стоящие на современный момент в этой области, дают основание считать это научное направление актуальным, требующим новых эффективных результатов.

Библиографический список

1. *Мальцев А.И.* Алгоритмы и рекурсивные функции. – М.: Наука, 1986. – 368 с.
2. *Kleene S.C.* General recursive functions of natural numbers. – Math. Ann., 1936, 112. P.727-742.
3. *Charch A.* An unsolvable problem of elementary number theory. – Amer.J.Math., 1936, 58. P. 345–363.
4. *Янов Ю.И.* О логических схемах алгоритмов. – В кн.: Проблемы кибернетики. Т.1. – М.: Наука, 1958. С. 75–109.
5. *Дейкстра Э.* Дисциплина программирования. – М.: Мир, 1978. – 275 с.
6. *Мазур М.* Качественная теория информации. – М.: Мир, 1974. – 238 с.
7. *Шнейдерман Б.* Психология программирования. Человеческие факторы в вычислительных и информационных системах. – М.: Радио и связь, 1984. – 304 с.
8. *Джонс М. Т.* Программирование искусственного интеллекта в приложениях/ М. Тим Джонс; пер. с англ. Осипов А. И. — М.: ДМК Пресс, 2006.
9. *Братко И.* Программирование на языке ПРОЛОГ для искусственного интеллекта. – М.: Мир, 1990. – 560 с.
10. *Джефф Рэшка, Элфрид Дастин, Джон Пол* Тестирование программного обеспечения. — М.: Лори, 2012. – 568 с.
11. *Норенков И. П.* Основы автоматизированного проектирования: учеб. для вузов. — 4-е изд., перераб. и доп. — М.: Изд-во МГТУ им. Н. Э. Баумана, 2009. — 430 с.
12. *Грис Д.* Наука программирования. – М.: Мир, 1984. – 416 с.
13. *Непомнящий В.А. Рякин О.М.* Оптимимзирующие преобразования программ. М.: Радио и связь, 1988. – 256 с.
14. *Дал У., Дейкстра Э., Хоор К.* Структурное программирование. – М.: Мир, 1975. – 247 с.
15. *Каширин И.Ю.* Алгебраический подход к исследованию программного инструментария сетей ЭВМ // Электросвязь. – 1996. – С. 14–17 .
16. *Kashirin I.Yu.* An Algebraic Approach to Object-Oriented Systems Design // Intern. Confer. "Problems of Mathematic and Informatic", HSU, Homel, 1994. – P. 47.
17. *Кострикин А.И.* Введение в алгебру. Основы алгебры. – М.: Физматлит, 1994. – 320 с.
18. *Тей А. и др.* Логический подход к искусственному интеллекту: от классической логики к логическому программированию. – М.: Мир, 1990. – 32 с.
19. *Цейтлин Г.Е.* Формальные аспекты структурного программирования с goto. – Программирование, 1984. N 1. – С. 3–16.
20. *Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л.* Алгебра. Языки. Программирование. – Киев, Наукова думка, 1974. – 328 с.
21. *Каширин И.Ю.* Объектно-ориентированное проектирование программ в среде С++. Вопросы практики и теории. М.: Госкомвуз России, “НИЦ-ПрИС”, 1996. – 192 с.
22. *Mosses P.* A basic abstract semantic algebra. – Lecture Notes of Computer Sciences, 1984. 173. P. 87–107.
23. *Беляева Н.П., Дмитриев М.Г.* Компьютерная алгебра в системах управления: некоторые проблемы и постановки. В кн.: “Теория и прикладные основы программных систем “. – РАН Институт программных систем. – Переславль-Залесский, 1994. – С. 265–280.
24. *Winkowski J., Maggiolo-Schettini A.* An Algebra of Processes. – Journal of Computer and System Sciences, 1987. – N 35. – P. 206 – 228.
25. *Bergstra J.A., Klop J.W.* Proving program inclusion using Hoare’s logic. – Theoretical computer science, v.30, 1984. – N 1. – P.1 – 48.
26. *Чернобров Л.В.* Верификация класса циклических программ без использования инвариантных циклов. – Программирование, 1984. – N 2. С. 3–14.
27. *Абрамов С.А.* Элементы анализа программ. – М.: Наука, 1986. – 368 с.
28. *Hehner E.C.R.* Predicative Programming Part I, Part II. – Communications of the ACM, v27. 1984. – N2. P. 134–143, 144–151.
29. *Криницкий Н.А.* Абстрактно-алгебраическая структура аналитической теории алгоритмов. – Программирование, 1984. N 5. С. 3–16.
30. *Backus J.* Can programming be liberated from von Neu-man style ? A functional style and its algebra of programs. – Communication of the ACM, v.21, 1978. N 8. P. 613–641.
31. *De Bakker J.W., Kok J.N., Meyer J.-J.Ch., Olderog E.-R., Zucker J.I.* Contrasting themes in the se-

mantics og imperative concurrency. – LNCS 224, Springer-Verlag, 1986. P. 51–121.

32. *Fisher M.J.,Ladner R.E.* Propositional dynamic logic of regular programs. – J. Comput. Systems Sci., vol.18. 1979. N 3. P. 194–211.

33. *Шилов Н.В.* О квазидетерминированной логике процессов. – В кн: Математическая теория программирования / под ред. Ершова А.П., Скорева Д.: ВЦ СО АН СССР, Новосибирск, 1985. – С. 65-81.

34. *Thorelli L.-E.* A linker allowing hierarchic composition of programms. – "Inf.Process'83: Proc. IFIP 9-th World Comput. Congr., Paris, Sept. 19–23, 1983", Amsterdam e.a., 1983. P.101–106.

35. *Nata G., Orefice S., Pocini G., Ruggiero, Tortora G. Legality* concepts for three-valued logic programs. –Theoretical Computer Science, v.120, 1993. N1. P. 45–68.

36. *P. Cousot, R. Cousot* "A la Burstall" intermitent assertions induction principles for proving inevitability properties of programs. – Software Practice & Experience, v.24. 1994. N 3. P. 123–168.

37. *Marimoto K., Iriguchi H.* A method of Compressing Tree Structures. – Parallel Computing, 1984. N 24. P. 265–288.

38. *Галушкин А.И., Кирсанова Д.В., Степанова М.В.* Нейронные ЭВМ. – М.: Радио и связь, 1997. – 240 с.

39. *Оленев Н.Н.* Основы параллельного программирования в системе MPI. — М.: ВЦ РАН, 2005. — 80 с.

40. *Wang Y., Parnas D.L.* Simulating the behavior of software modules by trace rewriting. – IEEE Trans.Software Eng, v.20, 1994. N 10. P. 750–759.

41. *Marriot K., Sondergaard H., Jones N.D.* Dennotational abstract innterpretation of logic programs. –ACM Trans.Programm. Lang. and Syst, v.16, 1994. N3. P. 607 – 648.

42. *Kashirin I.Yu., Korichnev L.P.* The Use of Program Semantic for Unifiable Object Representation in Object-Oriented Design. – Intern. Conf. of Math. & Inf. Probl. , GGU, Homel, 1994. – P. 12–14.

43. *Черноножкин С.К.* Меры сложности программ (обзор). – Системная информатика.: Новосибирск, Наука, 1996. С. 14–22.

44. *Tomita M.* An efficient augmented-context-free parsing algorithm. – Comput.Linguist., 1987. N 13. P. 31-46.

45. *Dwork C., Kannelakis P.C., Mitchell J.C.* On the sequential nature of unification. – J.Logic Program., 1984. N 1. P. 35-50.

46. *Каширин Д.И., Каширин И.Ю., Пылькин А.Н.* Полиморфическое представление знаний в Semsntic Web. – М., Горячая линия – Телеком, 2009. –136 с.

УДК 621.396

Ю.В. Петров

ГЕНЕРАТОР ПОСЛЕДОВАТЕЛЬНОСТЕЙ ПСЕВДОСЛУЧАЙНЫХ ЧИСЕЛ ЗАДАННОЙ РАЗМЕРНОСТИ С ПРАКТИЧЕСКИ РАВНОМЕРНЫМ РАСПРЕДЕЛЕНИЕМ

Предложен метод и программа моделирования последовательностей псевдослучайных равномерно распределенных чисел заданной размерности. Особенностью метода является формирование псевдослучайных чисел с использованием не безусловной вероятности (как в стандартных датчиках псевдослучайных чисел), а условной вероятности. Это позволяет получить практически равномерное распределение плотности распределения вероятности сгенерированных случайных чисел, моделирование которых на каждом шаге сводится к двум этапам: моделированию дискретной случайной величины с условной вероятностью (выбор интервала) и непосредственно формированию случайного числа, равномерно распределенного внутри выбранного интервала.

***Ключевые слова**: псевдослучайные числа, равномерная плотность распределения вероятности, гистограмма, дискретная случайная величина, условная вероятность.*

Введение. При исследовании сложных информационных систем методами имитационного компьютерного моделирования часто требуется генерировать последовательности случайных величин с различными законами плотности распределения вероятности заданной размерности [1]. Исходным материалом при формировании таких величин служат равномерно распределен-

ные в интервале от *0* до *1* случайные числа, которые, как правило, вырабатываются на компьютерах встроенными датчиками [2]. Так как ошибки моделирования случайных величин с различными законами распределения зависят от качества этих датчиков, то предъявляются особенно высокие требования к «равномерности» гистограммы распределения сгенерированных чисел и их независимости.

В настоящее время для формирования равномерно распределенных случайных чисел в основном используются программно реализованные генераторы псевдослучайных последовательностей, в основе которых лежат различные методы, такие как мультипликативно-конгруэнтный метод, регистры сдвига, метод Фибоначчи, «вихрь Мерсенна» и другие [3,4]. Наравне с необходимостью генерировать воспроизводимые последовательности псевдослучайных чисел (ПСЧ) существует необходимость генерировать абсолютно случайные числа. Такие генераторы строятся из комбинации также генераторов псевдослучайных последовательностей и внешних источников энтропии, например таких, как шумы токов и напряжений, текущее время, размер свободной памяти, счётчик тактов процессора.

Однако качество работы генераторов как псевдослучайных последовательностей, так и абсолютно случайных чисел, не всегда удовлетворяет заданным требованиям. Как видно из рисунка 1, частота попадания в различные диапазоны значений случайной величины при гистограммировании может отличаться в два и более раз.

Целями работы являются:

- разработка метода и программы моделирования последовательностей псевдослучайных чисел заданной размерности с практически равномерной гистограммой плотности распределения вероятности;
- использование разработанного метода при генерации последовательностей случайных величин с различными законами плотности распределения вероятности заданной размерности;
- разработка метода и программы моделирования случайных векторов с многомерными функциями плотности распределения вероятности заданной размерности.

Теоретическая часть. Предлагается, во-первых, качество генератора ПСЧ оценивать по гистограмме распределения последовательности сгенерированных случайных чисел заданной размерности *N*. Количество интервалов разбиения при гистограммировании *L* может быть выбрано по одной из известных формул: Старджесса, Брукса - Каррузера или их модификаций и развитий [5-7]. Необходимо обеспечить равновероятность попадания случайных чисел в каждый из интервалов.

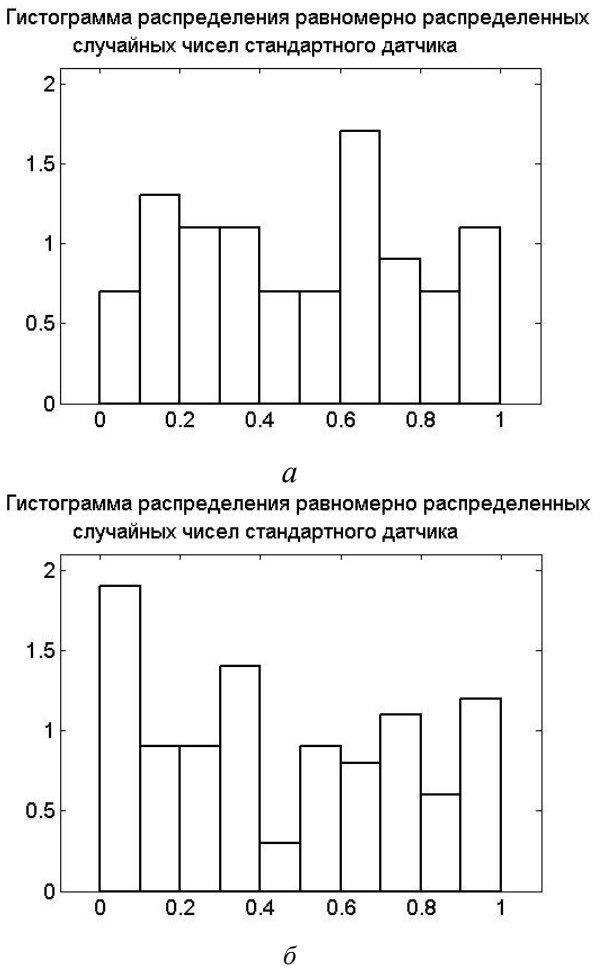


Рисунок 1 – Гистограммы распределения равномерно распределенных случайных величин, полученных с помощью встроенного датчика псевдослучайных чисел (две реализации - а и б)

Для этого, во-вторых, введем вспомогательную дискретную случайную величину $j = 1, L$, имеющую смысл номера интервала гистограммы.

Само моделирование случайных чисел на каждом шаге *i* сводится к двум этапам: моделированию дискретной случайной величины *j* с вероятностью p_{ji} (выбор интервала на *i*-ом шаге) и непосредственно формированию случайного числа, равномерно распределенного внутри выбранного интервала:

$$x_i = x_{0j} + \Delta x \cdot rand ,$$

где x_{0j} – левая граница *j*–го интервала l_j, $\Delta x = 1/L$ – размер интервала.

В третьих, предлагается на каждом *i*-ом шаге при формировании случайной величины *j* использовать не безусловную вероятность

$p_j = \frac{1}{L}$ (как в стандартных датчиках ПСЧ: вероятность попадания в любой интервал одинакова и постоянна), а условную вероятность $p_{ji/(i-1)}$, которую предлагается рассчитывать следующим образом:

$$p_{ji/(i-1)} = \begin{cases} p_{j(i-1)} \dfrac{N-i+1}{N-i} - \dfrac{1}{N-i}, x_{i-1} \in l_j \\ p_{j(i-1)} \dfrac{N-i+1}{N-i}, x_{i-1} \in l_j \end{cases}.$$

Таким образом, если на предыдущем шаге формирования случайное число попало в интервал j, то вероятность попадания в этот же интервал на следующем шаге $p_{ji/(i-1)}$ уменьшается. Вероятности попадания в другие интервалы увеличивается. «Начальная» вероятность попадания в каждый интервал одинакова $p_{j1/0} = \frac{1}{L}$.

Суть метода демонстрирует простой числовой пример генерации последовательности псевдослучайных равномерно распределенных от 0 до 1 чисел размерностью 10. Количество интервалов разбиения при гистограммировании равно 5. «Начальная» вероятность попадания в каждый интервал (в момент времени 0) равна 0,2 (рисунок 2).

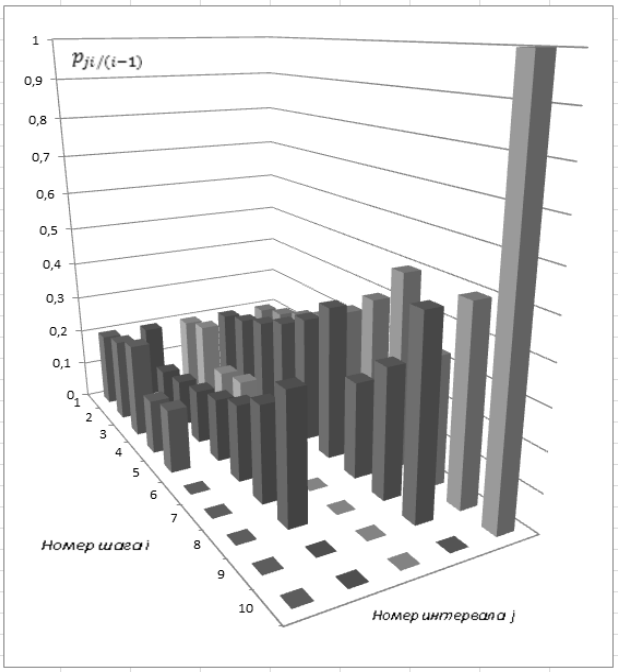


Рисунок 2 – Графики изменения во времени условных вероятностей попадания дискретной случайной величины *j* в один из пяти интервалов

Пусть на первом шаге случайным образом выбран интервал 2. Формирование случайного числа x_i, равномерно распределенного внутри выбранного интервала, производится в соответствии с приведенным выше выражением.

Условная вероятность на следующий шаг будет рассчитана: для этого интервала как 0,2*(10-1+1)/(10-1)-1/(10-1)=0,11, для остальных как 0,2*(10-1+1)/(10-1)=0,22. Отметим, что вероятность попадания в интервал 2 на следующем шаге уменьшилась (с 0,2 до 0,11), вероятность попадания в другие интервалы увеличилась (с 0,2 до 0,22). Пусть на втором шаге случайным образом выбран интервал 3. Условные вероятности при этом будут рассчитаны: 0,125 (для интервалов 2 и 3) и 0,25 (для остальных) соответственно.

В итоге получена, например, последовательность номеров интервалов, в которые попадала случайная величина {2,3,1,3,1,4,5,2,4,5}. Изменения во времени условных вероятностей попадания дискретной случайной величины j в один из пяти интервалов для данного примера приведены на рисунке 2. Отметим, что в определенные моменты времени вероятности попадания в некоторые интервалы становятся равными нулю (например, для интервала 1 это происходит на 6 шаге), что означает, что в данном интервале уже сгенерировано «достаточно» случайных чисел и появление там еще одного числа вызовет превышение необходимого значения равномерной плотности распределения вероятности.

Определенный интерес вызывает «предельный» вариант выбора количества интервалов разбиения при гистограммировании L, когда оно равно размеру заданной реализации N. В этом случае в каждый интервал будет попадать всего по одному значению случайной величины. Это позволяет получать практически равномерную гистограмму распределения случайных чисел при любом значении количества интервалов разбиения при последующем гистограммировании.

Условная вероятность $p_{ji/(i-1)}$ в этом варианте будет принимать всего два значения: $1/(N-i)$ или 0. «Начальная» вероятность попадания в каждый интервал будет, как и ранее, одинакова $p_{j1/0} = \frac{1}{L} = \frac{1}{N}$. Случайное число, вырабатываемое на каждом шаге, как и ранее, равномерно распределено внутри выбранного интервала.

Экспериментальные исследования. Для экспериментальной проверки предложенного метода моделирования была разработана программа-генератор последовательностей псевдослучайных равномерно распределенных чисел заданной размерности. Оценка проводилась при различных значениях размерностей и количествах

интервалов разбиения при гистограммировании.

В качестве примера на рисунке 3 приведены гистограммы распределения случайных величин, полученных с помощью стандартного встроенного и предлагаемого датчиков равномерно распределенных псевдослучайных чисел.

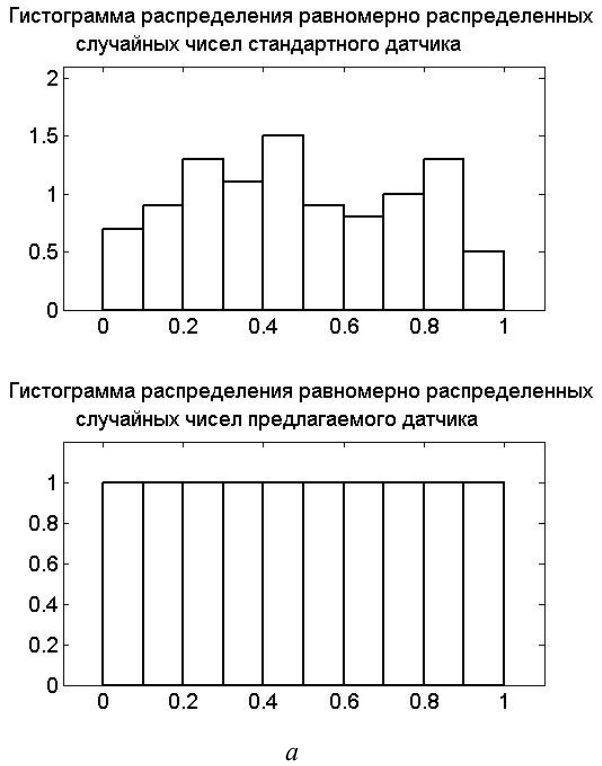


Рисунок 3 – Гистограммы распределения равномерно распределенных случайных величин, полученных с помощью встроенного (а) и предлагаемого (б) датчиков псевдослучайных чисел

С целью определения «независимости» (некоррелированности) генерируемых случайных чисел были проведены соответствующие исследования. Для всех генерируемых последовательностей рассчитывались их статистические характеристики и определялись автокорреляционные функции. Для примера на рисунке 4 приведена одна из нормированных автокорреляционных функций *r(k)* последовательности равномерно распределенных случайных величин, полученных с помощью предлагаемого датчика.

Проведенные исследования показали работоспособность метода.

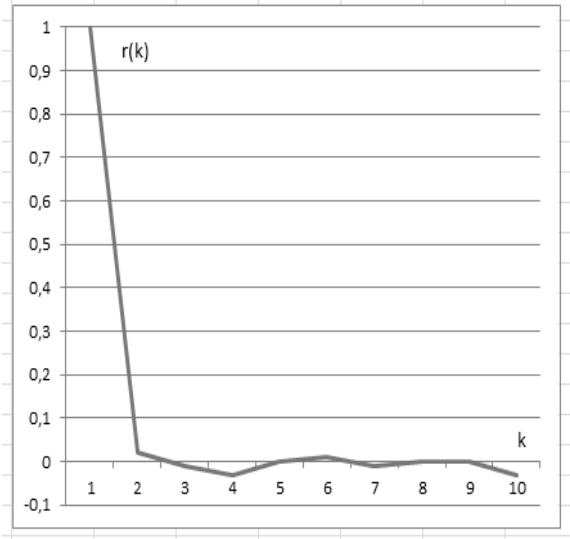


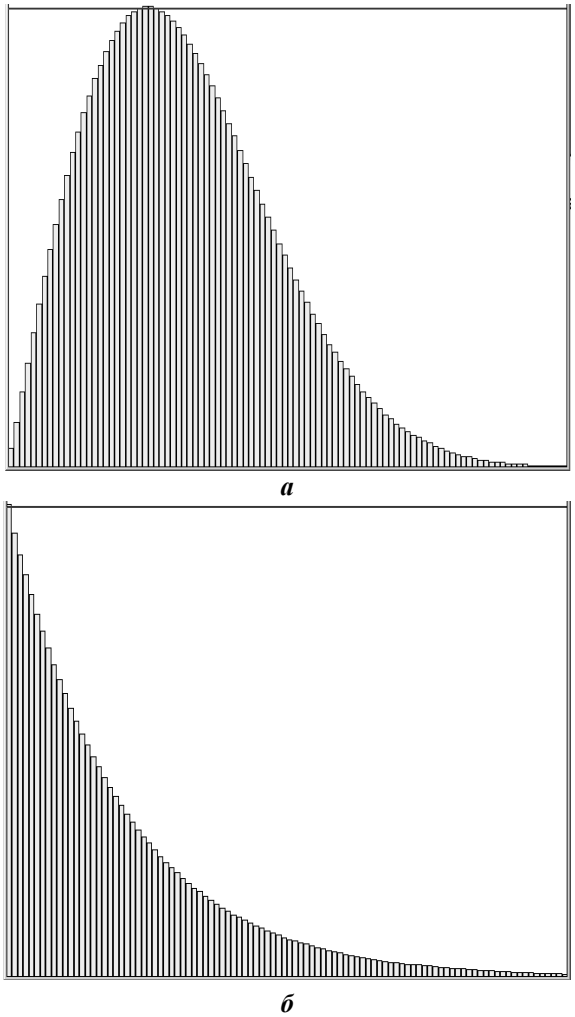
Рисунок 4 – Нормированная автокорреляционная функция r(k) последовательности равномерно распределенных случайных величин, полученных с помощью предлагаемого датчика

Использование генерируемых последовательностей в качестве «исходного материала» при формировании последовательности случайных величин с различными законами плотности распределения вероятности заданной размерности позволило значительно повысить их качество. При сравнении «сгенерированных» и «заданных» законов использовался критерий согласия хи-квадрат Пирсона. Вычисленные значения статистики не превосходили квантиль распределения хи-квадрат для уровня значимости 0,9 в 100 экспериментах из 100 проведенных (у «стандартных» датчиков – только в 15 экспериментах из 100).

В качестве примера на рисунке 5 приведены гистограммы распределения случайных чисел с релеевским и экспоненциальным законами плотности распределения вероятности, полученные с помощью предлагаемого датчика.

Предлагаемый метод позволяет моделировать также многомерные случайные величины (случайные вектора) с многомерными функциями плотности распределения вероятности

$W(x_1,\ldots,x_N)$. Идея метода такая же, как и в одномерном случае, с той разницей, что здесь моделируются случайные точки, равномерно распределённые не на плоскости, а в *N*-мерном пространстве.



а

б

Рисунок 5 – Гистограммы распределения случайных чисел с релеевским (а) и экспоненциальным (б) законами плотности распределения вероятности, полученные с помощью предлагаемого датчика

Количество интервалов разбиения при гистограммировании по каждой оси *N*-мерного пространства $x_1,\ldots,x_N$ может быть выбрано, как и ранее, по одной из известных формул. В общем случае количество интервалов разбиения по каждой оси может быть разным. Необходимо обеспечить равновероятность попадания случайных чисел в каждый из интервалов.

Введем *N* вспомогательных дискретных случайных величин $j_1,\ldots,j_N$, $j_n = 1, L_n$, имеющих смысл номеров интервалов по соответствующим осям.

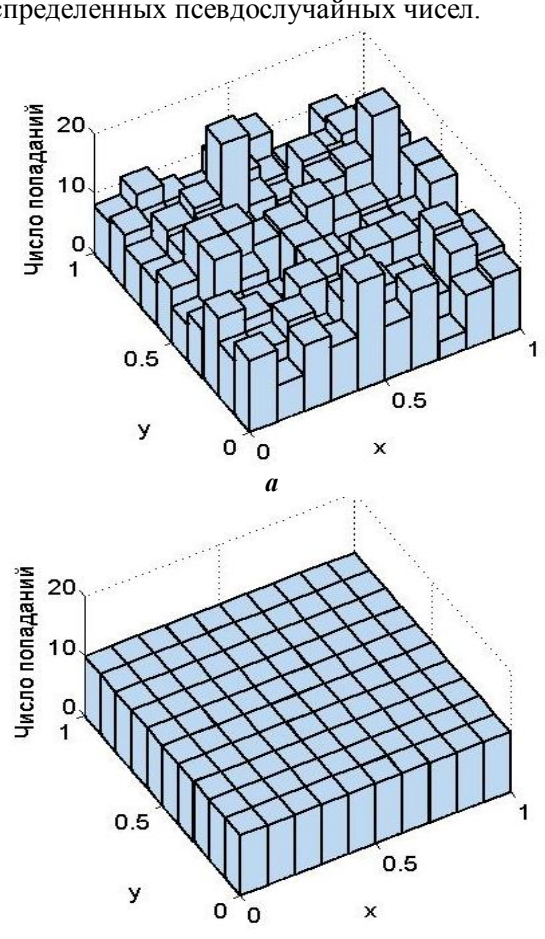
Моделирование случайных векторов сводится к двум этапам: моделированию дискретных случайных величин $j_1,\ldots,j_N$ с вероятностями p_{ij_n} (выбор интервалов по каждой *n*-й оси на *i*-м шаге) и непосредственно формированию случайных координат вектора, равномерно распределенных внутри выбранных интервалов:

$$x_{ni} = x_{0\,jn} + \Delta x_n \cdot rand_{ni},$$

где $x_{0\,jn}$ – левая граница j_n-го интервала, $\Delta x_n = \dfrac{1}{L_n}$ – размер соответствующего интервала.

На каждом *i*-м шаге формирования случайного вектора используется не безусловная вероятность, а условная вероятность $p_{nji/(i-1)}$, которая рассчитывается аналогично вышеописанному.

В качестве примера на рисунке 6 приведены гистограммы распределения координат двумерного случайного вектора с равномерной по двум осям плотностью распределения вероятности $W(x,y)$, сформированного с помощью стандартного и предлагаемого датчиков равномерно распределенных псевдослучайных чисел.



а

б

Рисунок 6 – Гистограммы распределения равномерно распределенных координат двумерного случайного вектора, полученных с помощью встроенного (а) и предлагаемого (б) датчиков псевдослучайных чисел

Заключение. Предложен метод и программа моделирования последовательностей псевдослучайных чисел заданной размерности, позволяющие получать равномерное распределение плотности распределения вероятности сгенерированных случайных чисел, удовлетворяющее критерию согласия хи-квадрат с уровнем значимости 0,9 и выше.

Использование генерируемых последовательностей в качестве «исходного материала» при формировании последовательности случайных величин с различными законами плотности распределения вероятности заданной размерности позволило значительно повысить их качество. Значения статистики не превосходили квантиль распределения хи-квадрат для уровня значимости 0,9 в 100 экспериментах из 100 проведенных (у «стандартных» датчиков – только в 15 экспериментах из 100).

Предлагаемый метод позволяет моделировать также многомерные случайные величины (случайные вектора) с многомерными функциями плотности распределения вероятности.

Библиографический список

1. *Строгалев В.П., Толкачева И.О.* Имитационное моделирование. МГТУ им. Баумана, 2008. ISBN 978-5-7038-3021-5.

2. Методы математического моделирования радиотехнических систем: под ред. Ю.В. Петрова: Балт. гос. техн. ун-т – СПб, 2005.

3. *Шалыгин А.С., Палагин Ю.И.* Прикладные методы статистического моделирования. – Л.: Машиностроение, Ленинград. отд-ние, 1986.

4. *Дональд Э. Кнут.* Искусство программирования. М.: Вильямс, 2000. - 832 с.- ISBN 5-8459-0081-6.

5. *Шторм Р.* Теория вероятностей. Математическая статистика. Статистический контроль качества М.: Мир, 1970.

6. *Таушанов З., Тонева Е., Пенова Р.* Вычисление энтропийного коэффициента при малых выборках // Изобретательство, стандартизация, качество, 1973. №5.

7. *Тонева Е.* Аппроксимация распределений погрешности средств измерений// Измерительная техника, 1981. №6. - С.15-16.